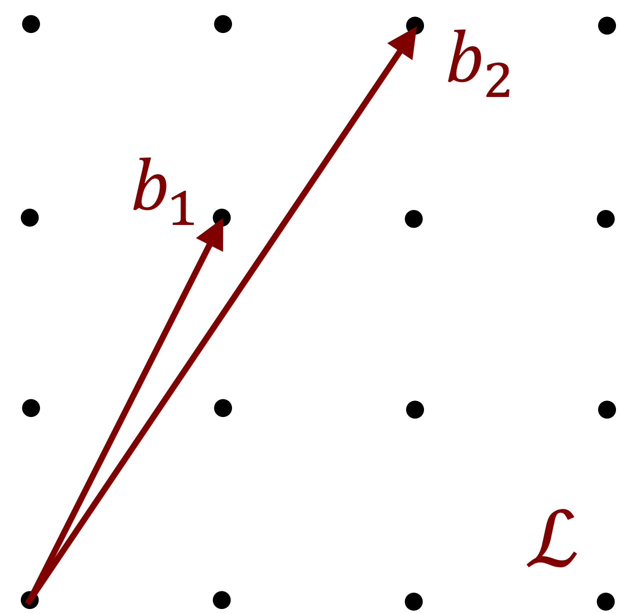
Solving SVP and CVP in 2^n Time Using Discrete Gaussian Sampling

Divesh Aggarwal
National University of Singapore (NUS)
Daniel Dadush
Centrum Wiskunde en Informatica (CWI)
Oded Regev
Noah Stephens-Davidowitz
New York University (NYU)

Lattices

A lattice $\mathcal{L} \subseteq \mathbb{R}^n$ is all integral combinations of some basis $\mathrm{B} = (b_1, \dots, b_n)$.

$\mathcal{L}(B)$ denotes lattice generated by B.

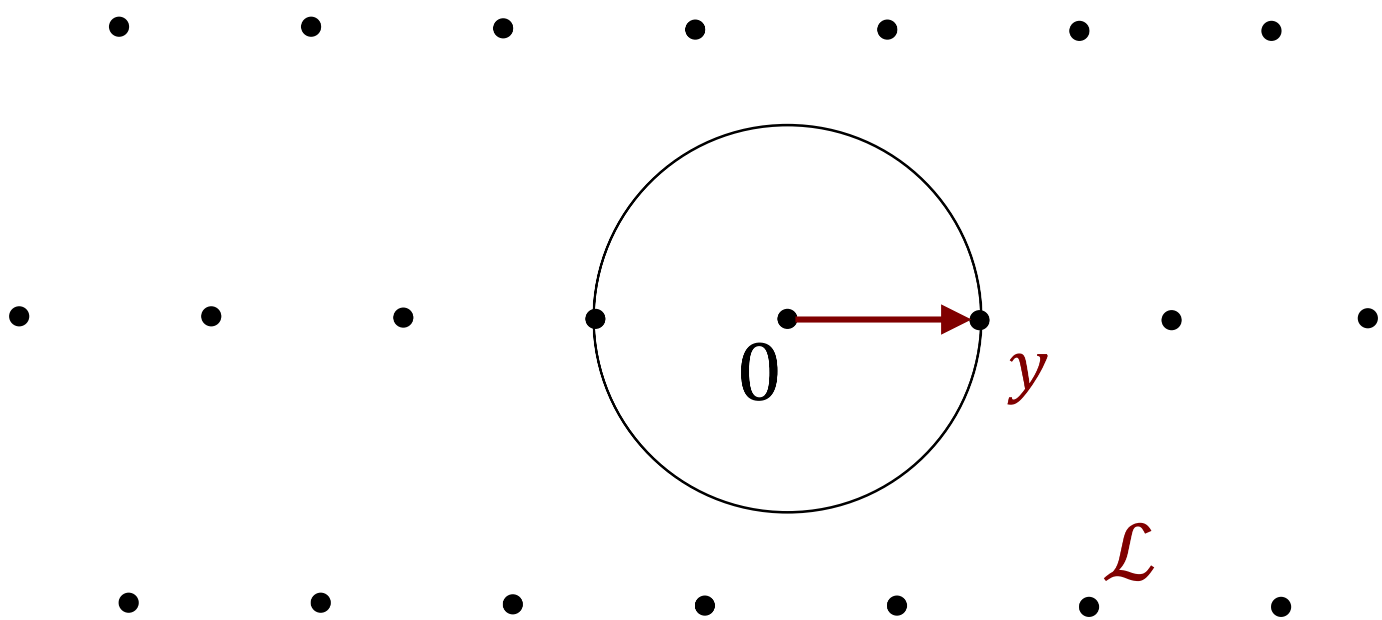


Act I: The Shortest Vector Problem

Shortest Vector Problem (SVP)

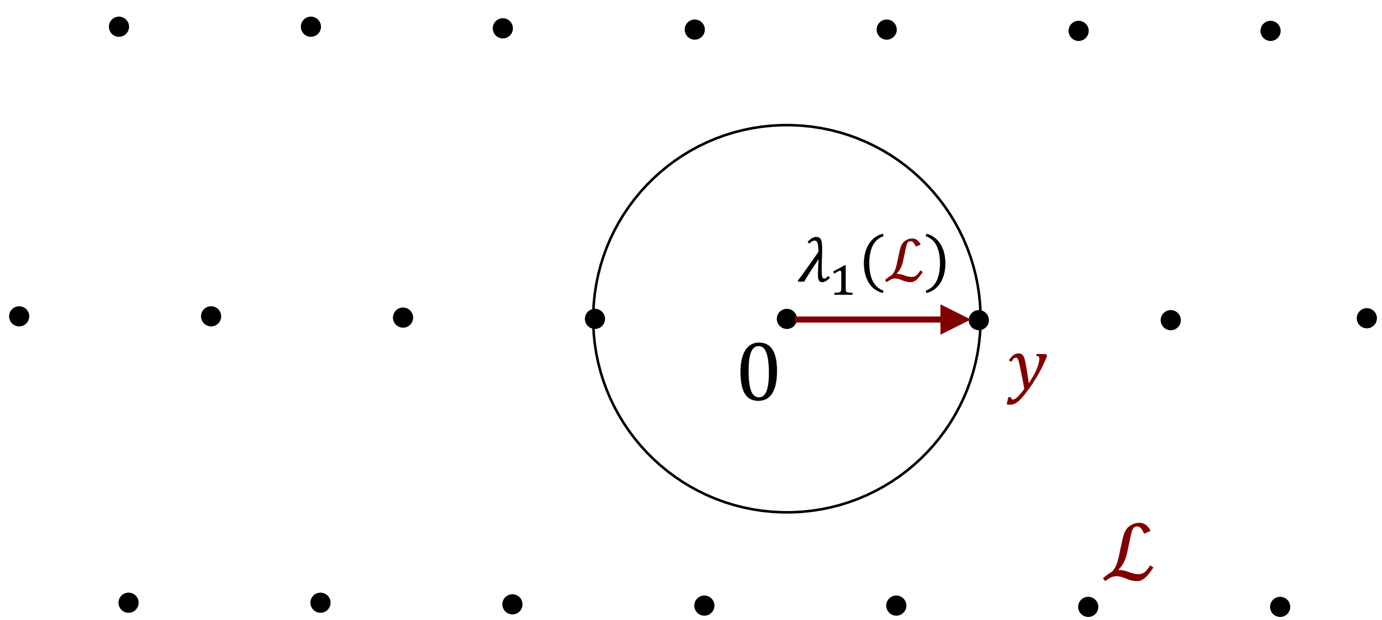
Given: Lattice basis $B \in \mathbb{Q}^{n \times n}$.

Goal: Compute shortest non-zero vector in $\mathcal{L}(B)$.



Shortest Vector Problem (SVP)

$\lambda_1(\mathcal{L})$ = length of shortest non-zero vector



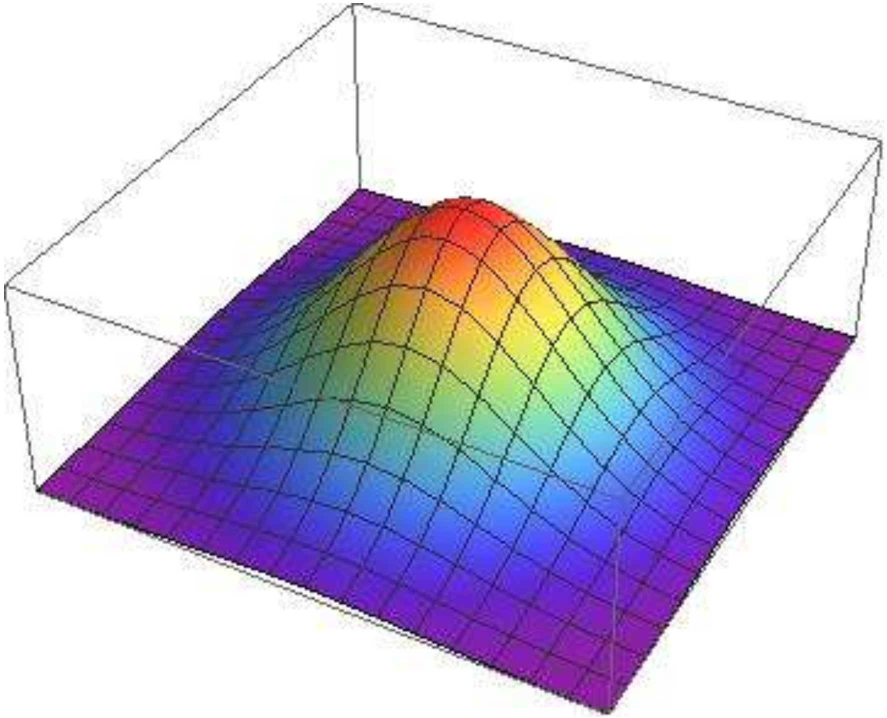
Algorithms for SVP

	Time	Space
[Kan86,HS07,MW15] (Enumeration)	$n^{O(n)}$	$\mathrm{poly}(n)$
[AKS01] (Sieving)	$2^{O(n)}$	$2^{O(n)}$
[NV08, PS09, MV10a, …]	$2^{2.465n+o(n)}$	$2^{1.233n+o(n)}$
[MV10b] (Voronoi cell, deterministic, CVP)	$2^{2n+o(n)}$	$2^{n+o(n)}$
[ADRS15]	$2^{n+o(n)}$	$2^{n+o(n)}$

Our Algorithm

Gaussian Distribution

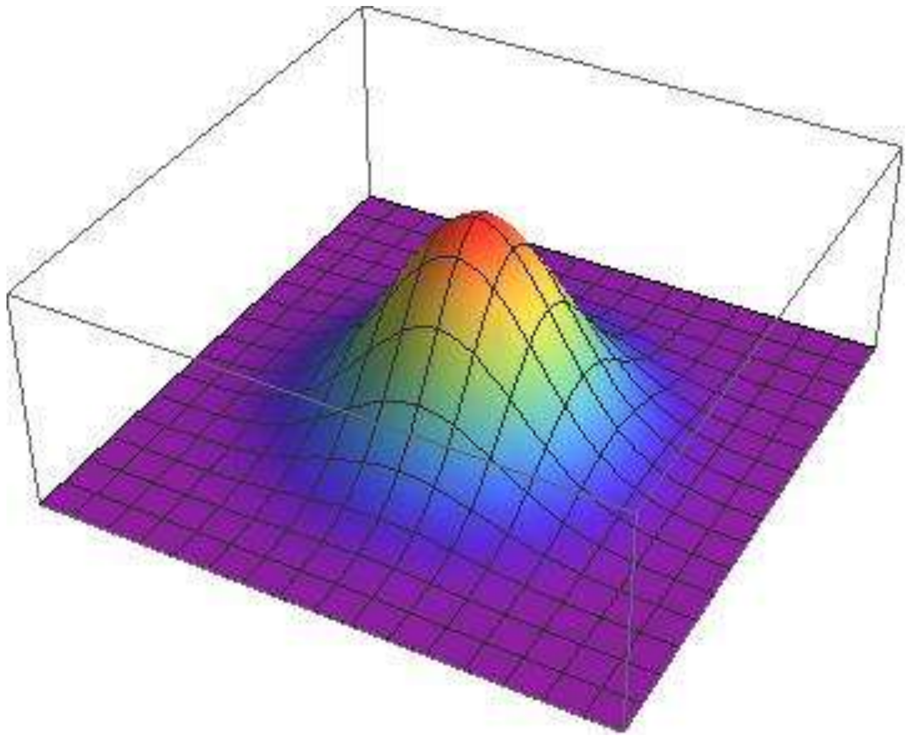
$$\mathsf{Gauss}(s) := \Pr[\mathbf{x}] \propto e^{-\|\mathbf{x}\|^2/s^2}$$



$s = 20$

Gaussian Distribution

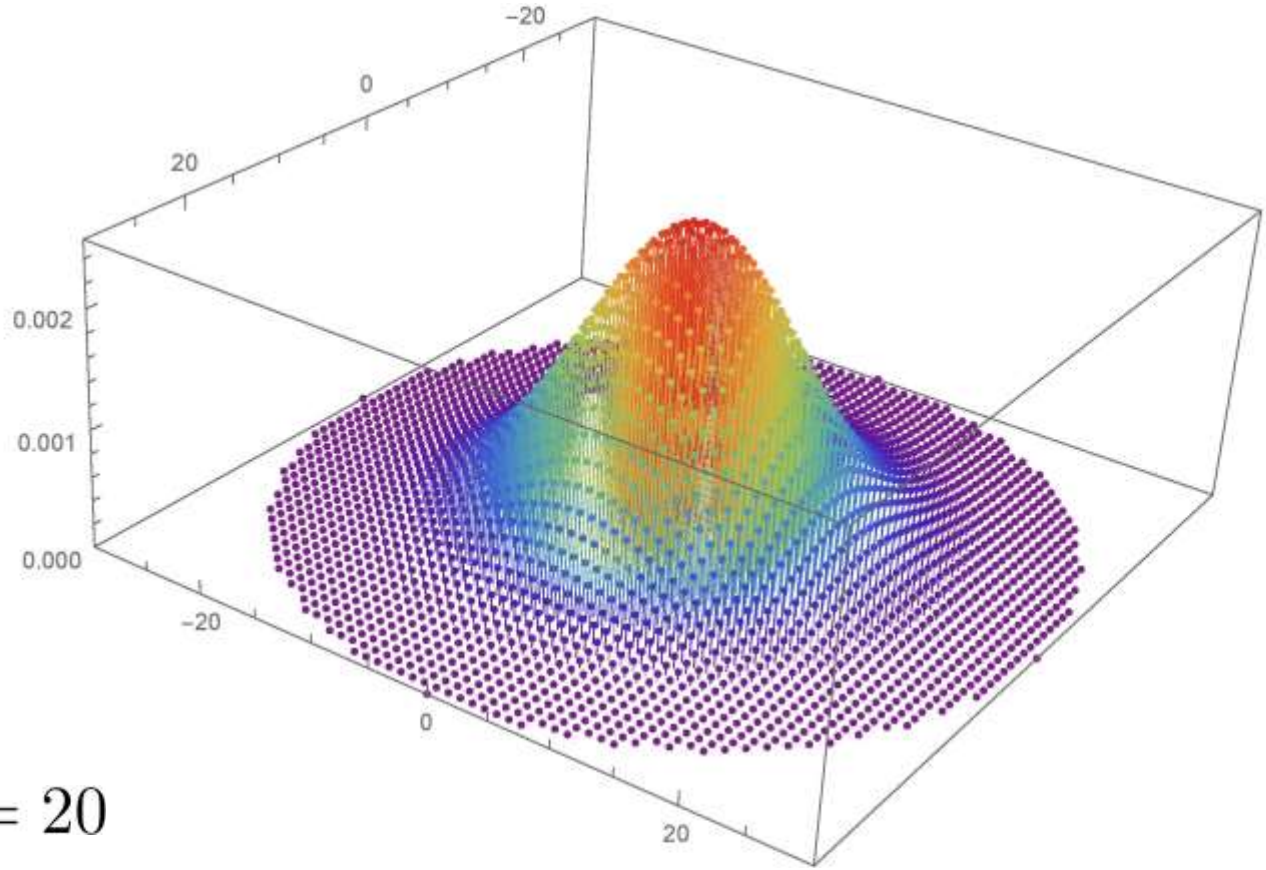
$$\mathsf{Gauss}(s) := \Pr[\mathbf{x}] \propto e^{-\|\mathbf{x}\|^2/s^2}$$



$s = 10$

Discrete Gaussian Distribution

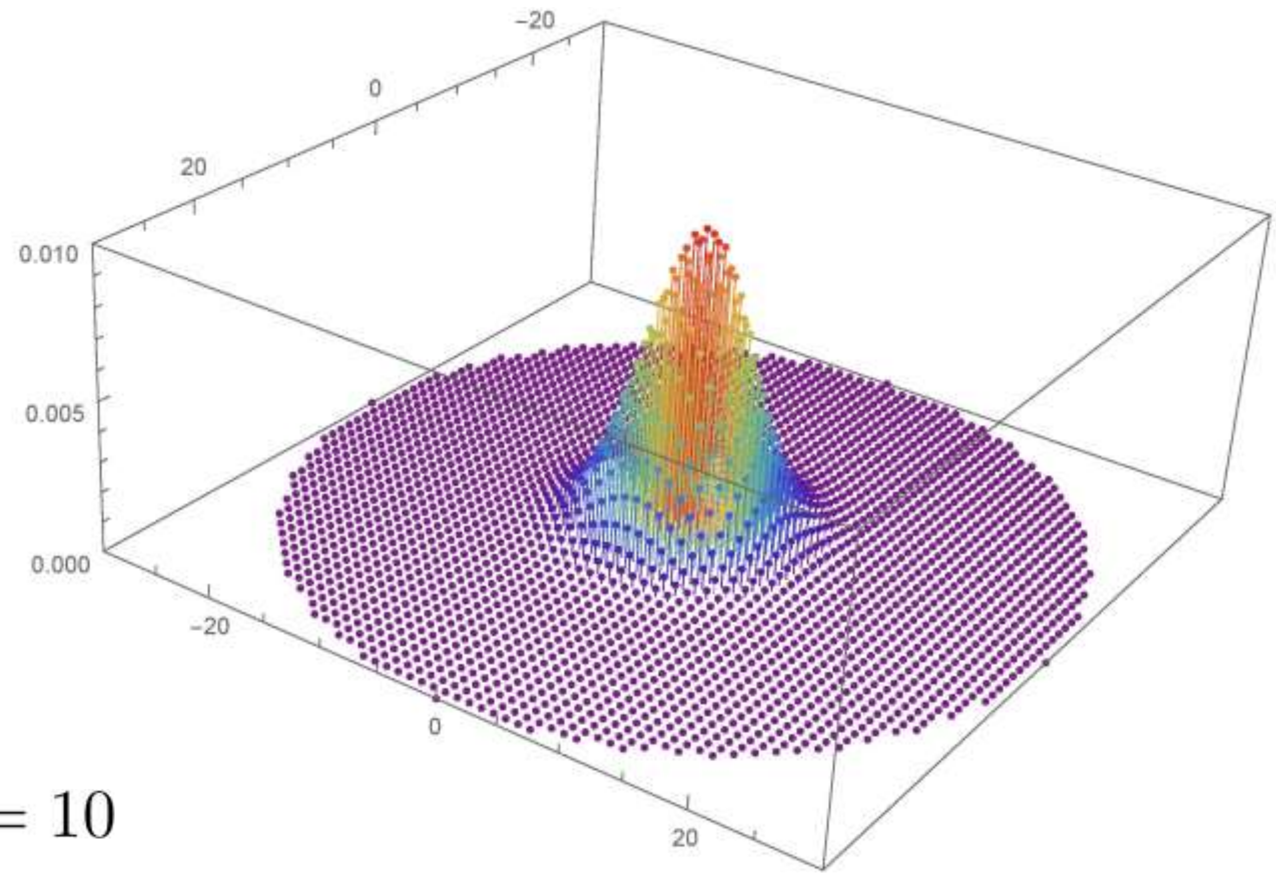
$$D_{\mathcal{L},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}\|^2/s^2}$$



$s = 20$

Discrete Gaussian Distribution

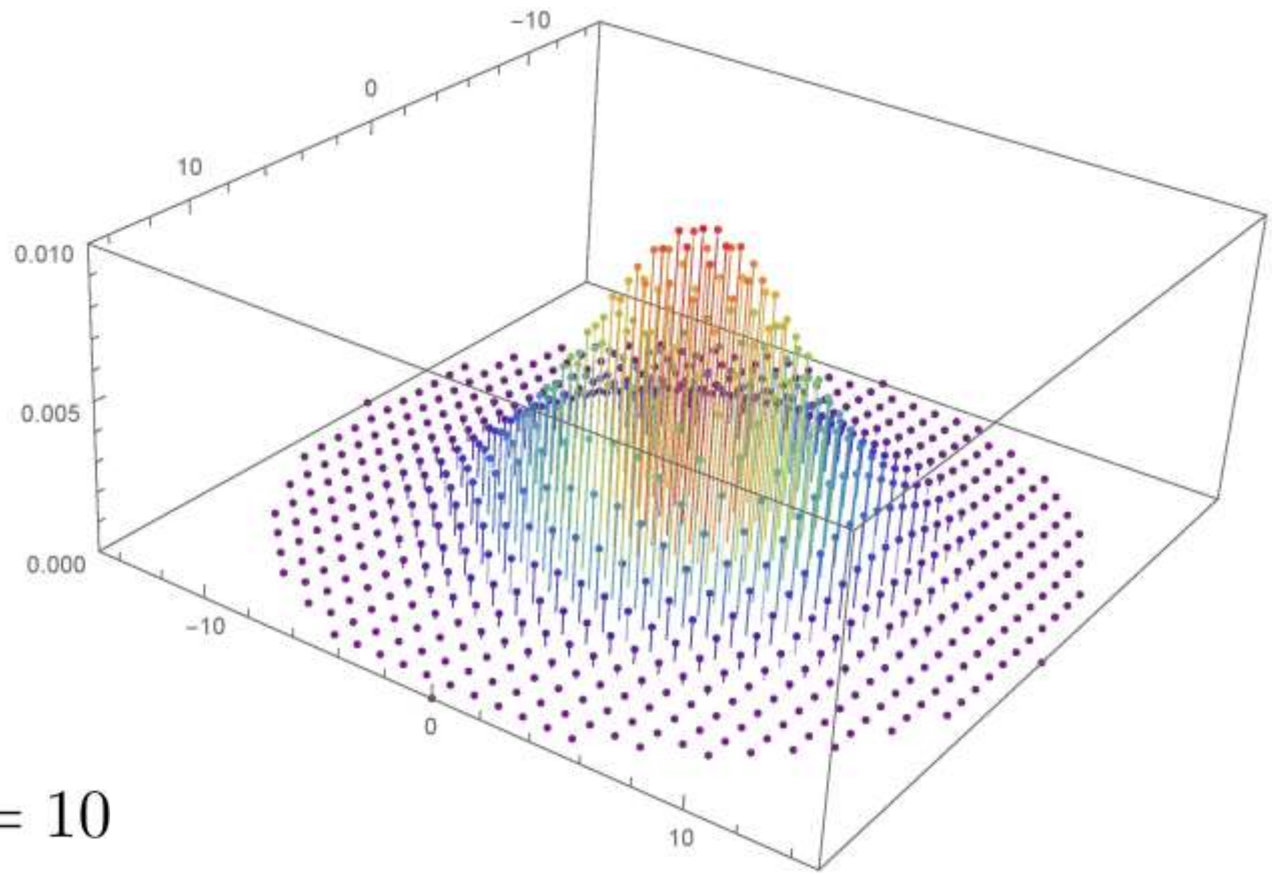
$$D_{\mathcal{L},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}\|^2/s^2}$$



$s = 10$

Discrete Gaussian Distribution

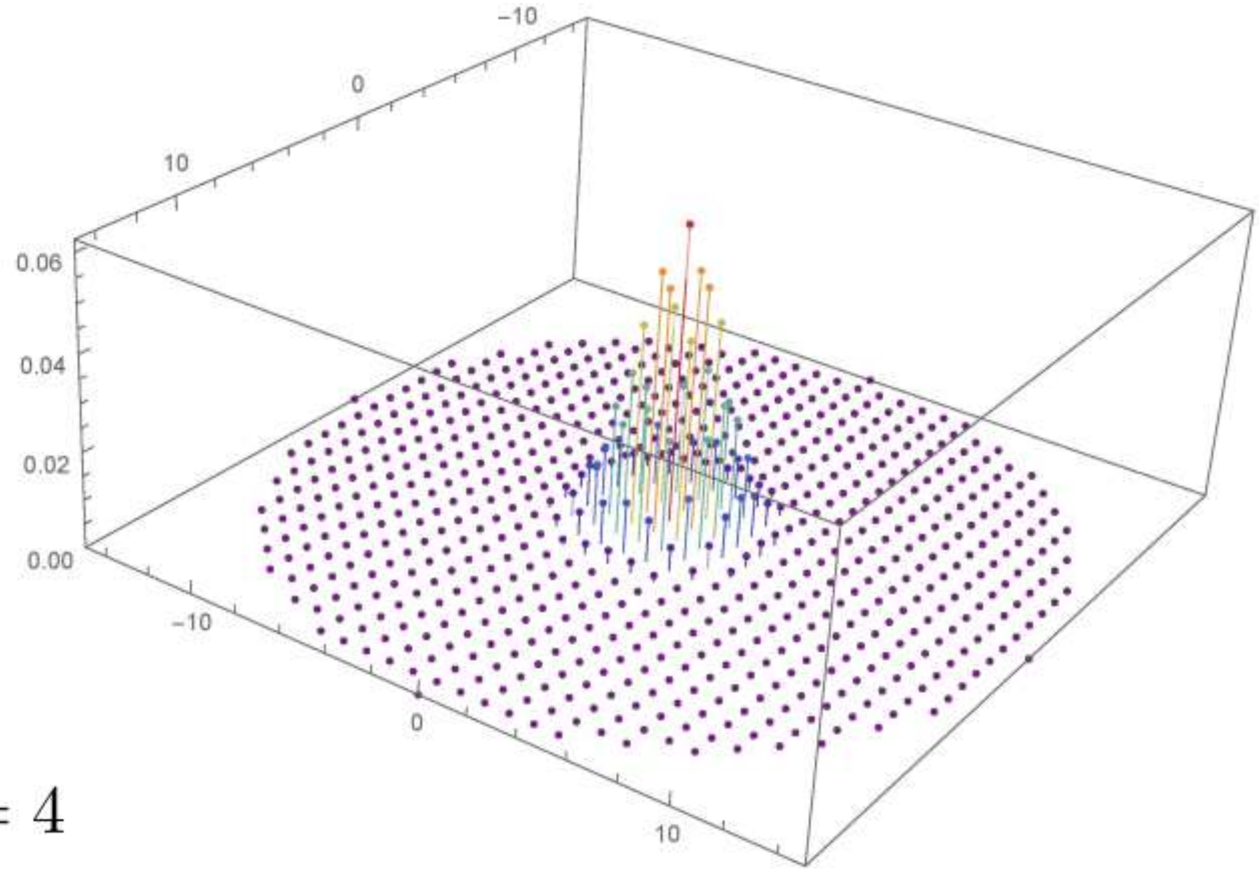
$$D_{\mathcal{L},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}\|^2/s^2}$$



$s = 10$

Discrete Gaussian Distribution

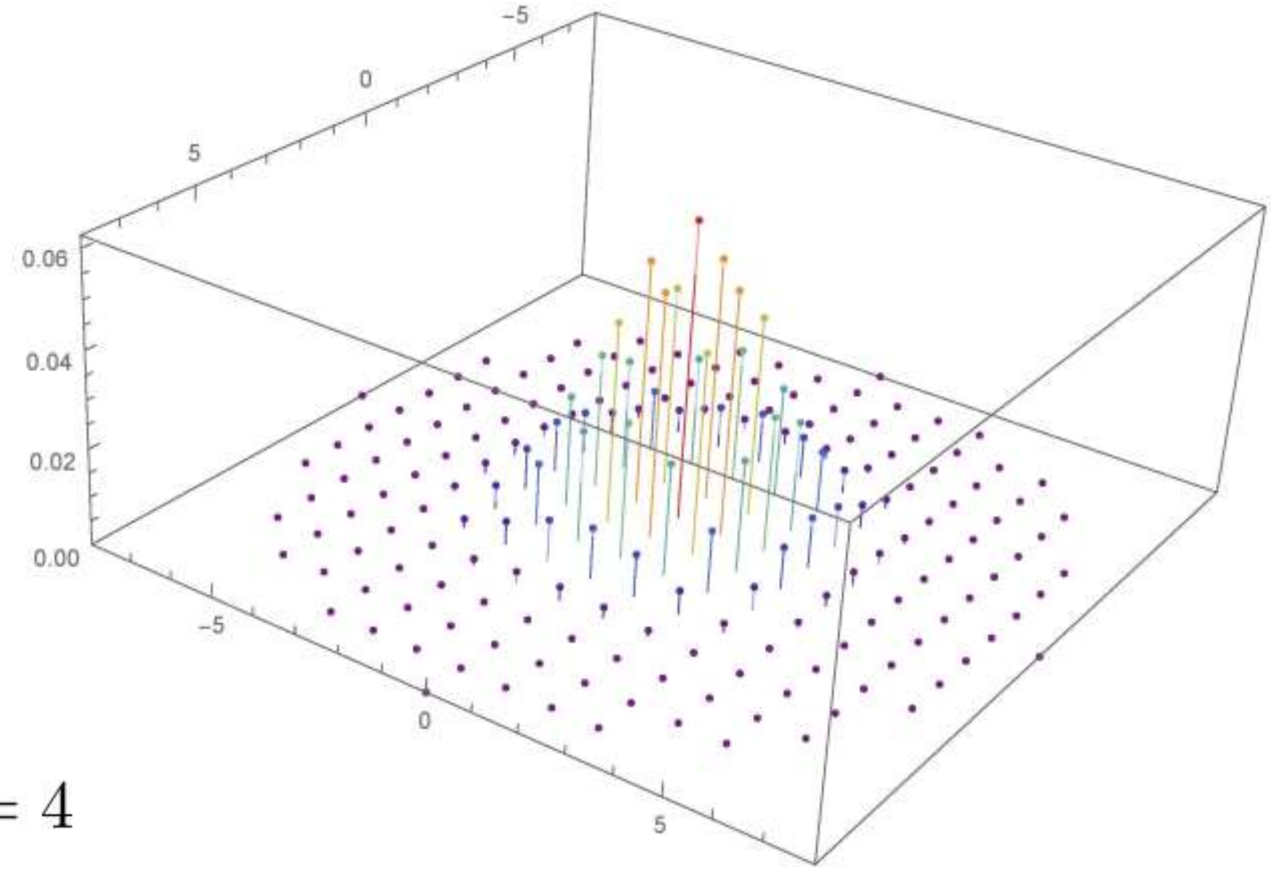
$$D_{\mathcal{L},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}\|^2/s^2}$$



$s = 4$

Discrete Gaussian Distribution

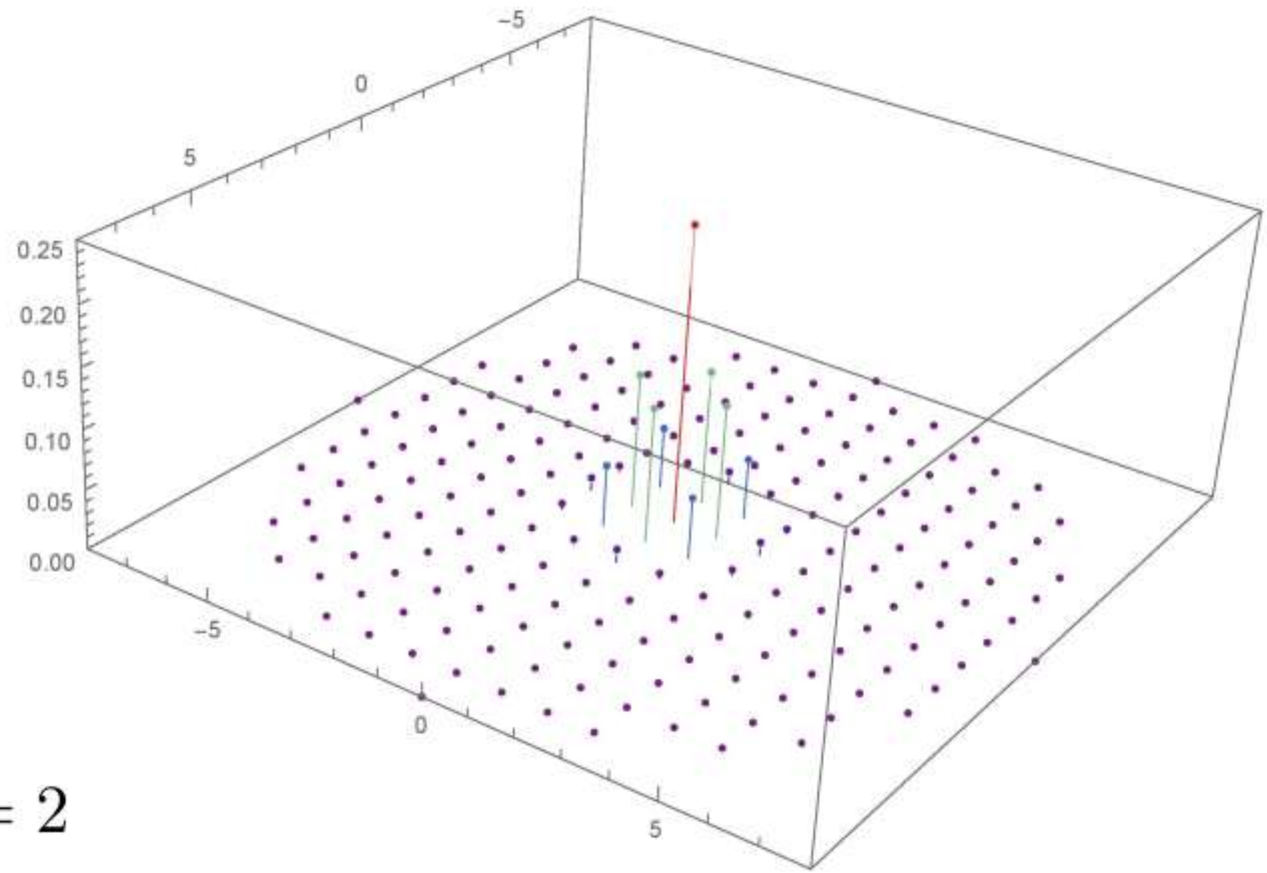
$$D_{\mathcal{L},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}\|^2/s^2}$$



$s = 4$

Discrete Gaussian Distribution

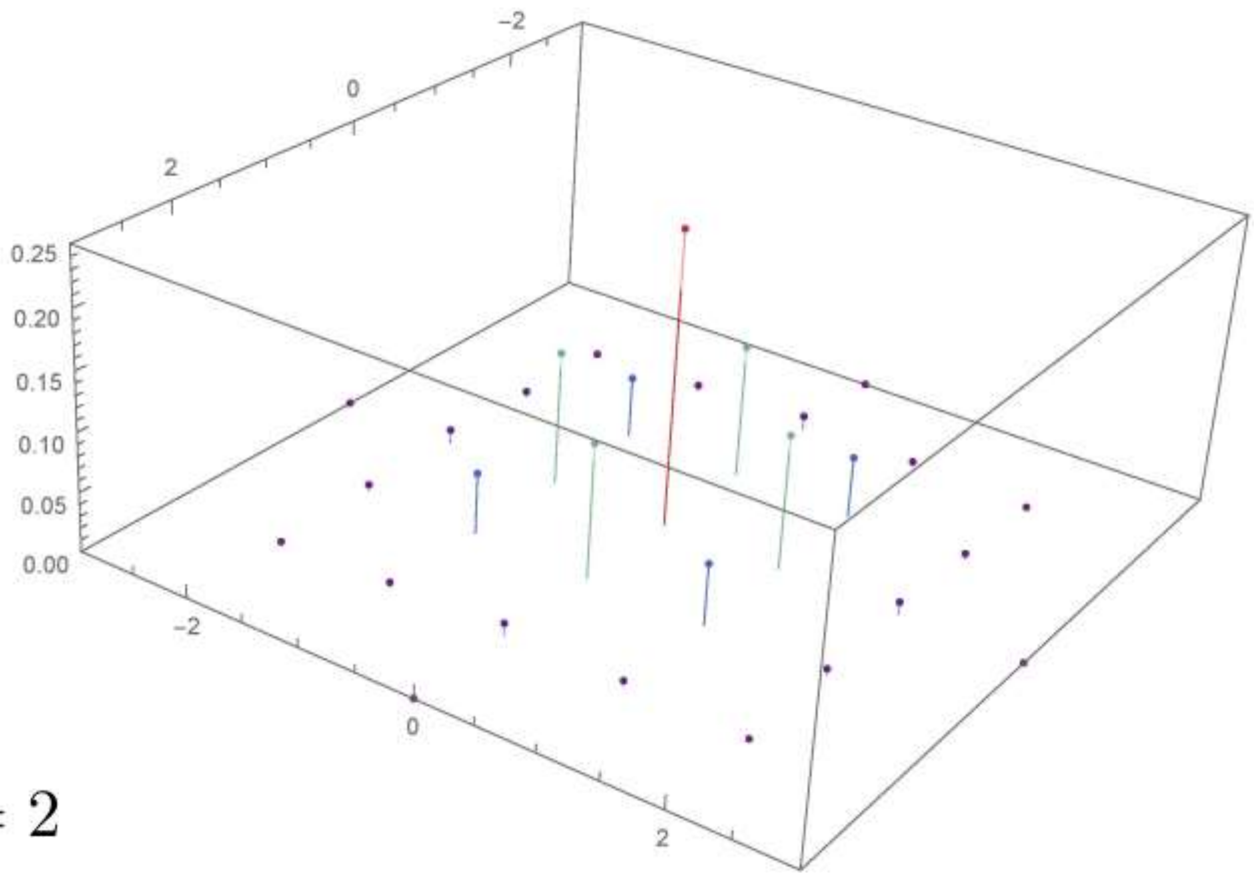
$$D_{\mathcal{L},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}\|^2/s^2}$$



$s = 2$

Discrete Gaussian Distribution

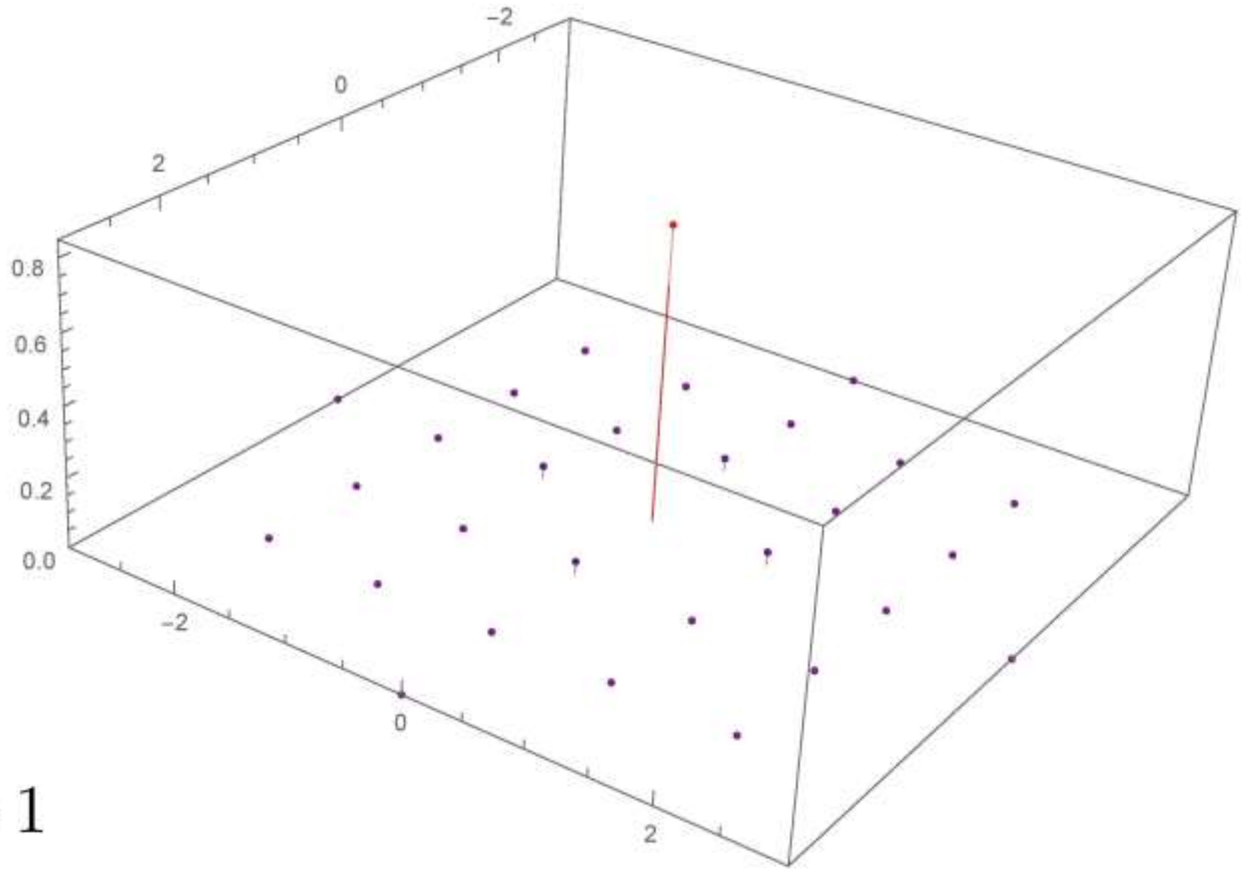
$$D_{\mathcal{L},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}\|^2/s^2}$$



$s = 2$

Discrete Gaussian Distribution

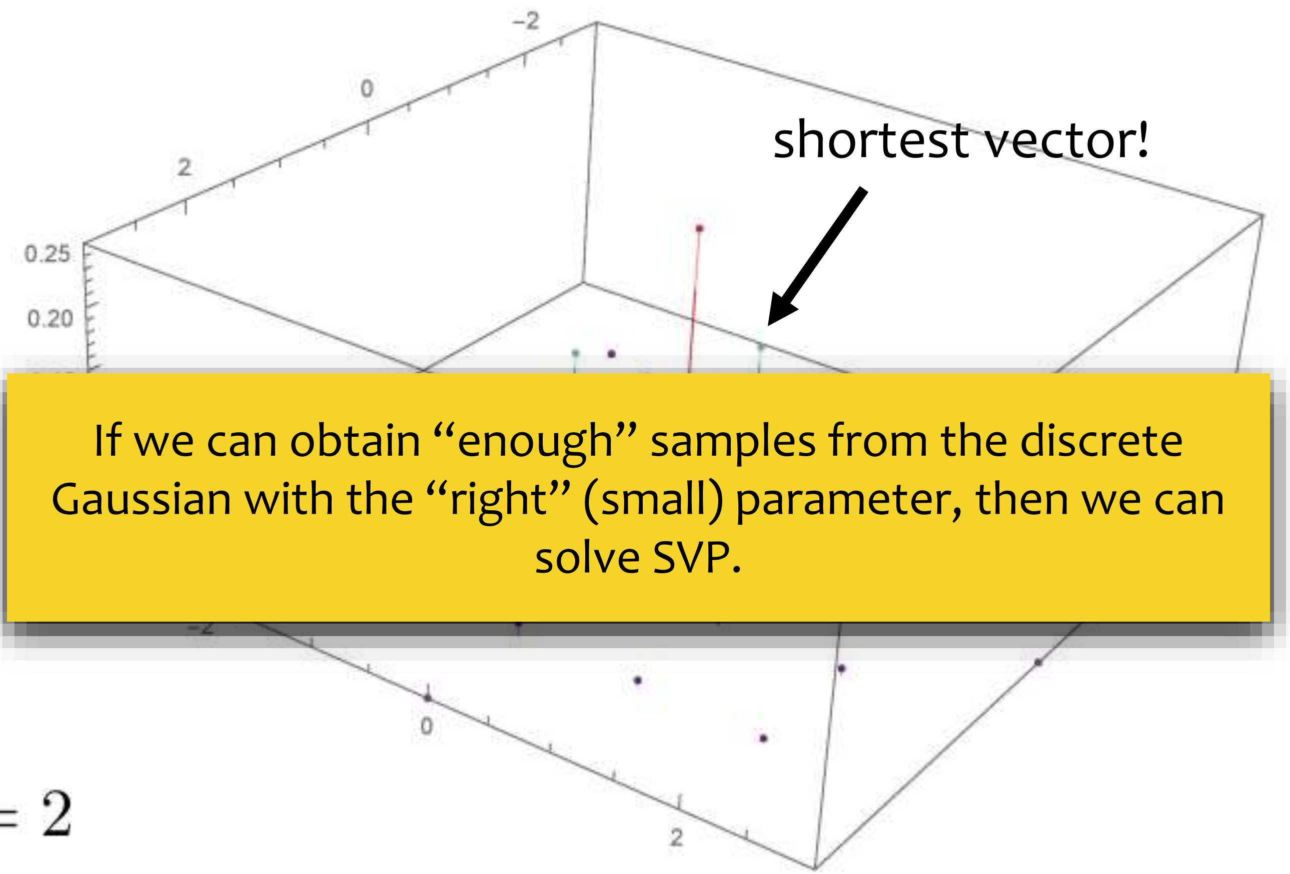
$$D_{\mathcal{L},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}\|^2/s^2}$$



$s = 1$

Discrete Gaussian Distribution

$$D_{\mathcal{L},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}\|^2/s^2}$$



If we can obtain "enough" samples from the discrete Gaussian with the "right" (small) parameter, then we can solve SVP.

$s = 2$

Discrete Gaussian Distribution

We need at most 1.38^n vectors with $s \approx \lambda_1(\mathcal{L})/\sqrt{n}$ [KL78].
(uses bounds on the kissing number)

$D_{\mathcal{L},s}$ is very well-studied for very high parameters, $s \gtrsim \lambda_n(\mathcal{L})$, i.e. above the "smoothing parameter" of the lattice.

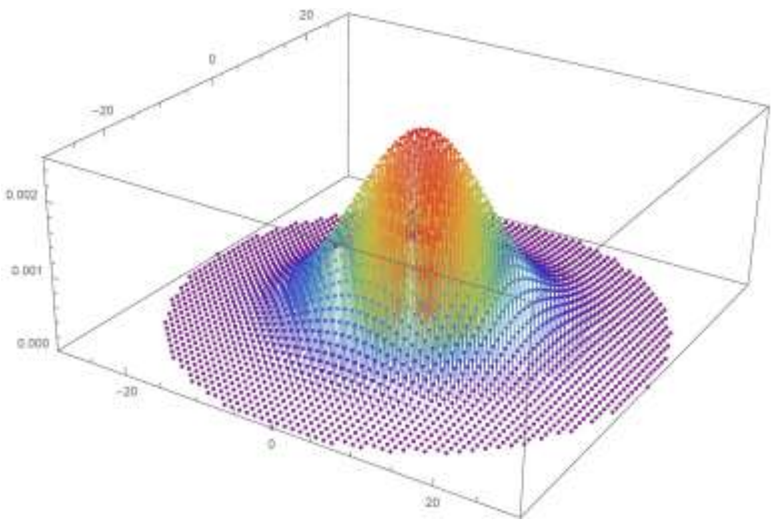
[Kle00, GPV08] show how to sample in this regime in polynomial time.

(Previously could not do much better, even in exponential time.)

Discrete Gaussian Distribution

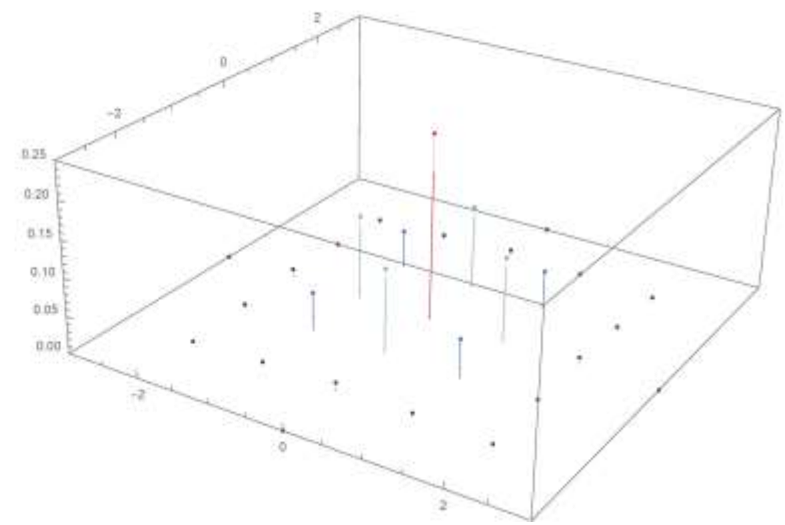
Easy

[Kle00, GPV08]



Our goal

Hard



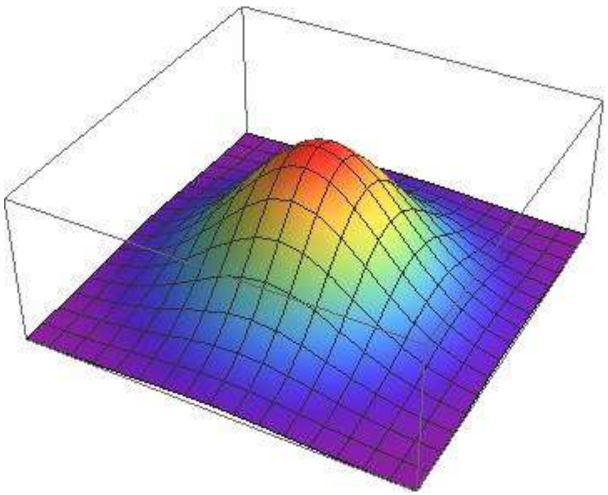
$s \gg \lambda_1(\mathcal{L})$

$s \approx \lambda_1(\mathcal{L})/\sqrt{n}$

Can we use samples from the LHS to get samples from the RHS?

Discrete Gaussian Distribution

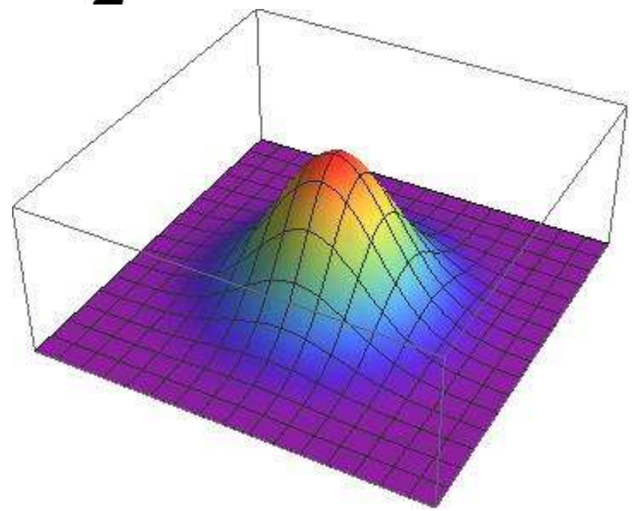
$$\mathbf{x} \sim \mathsf{Gauss}(s)$$



2

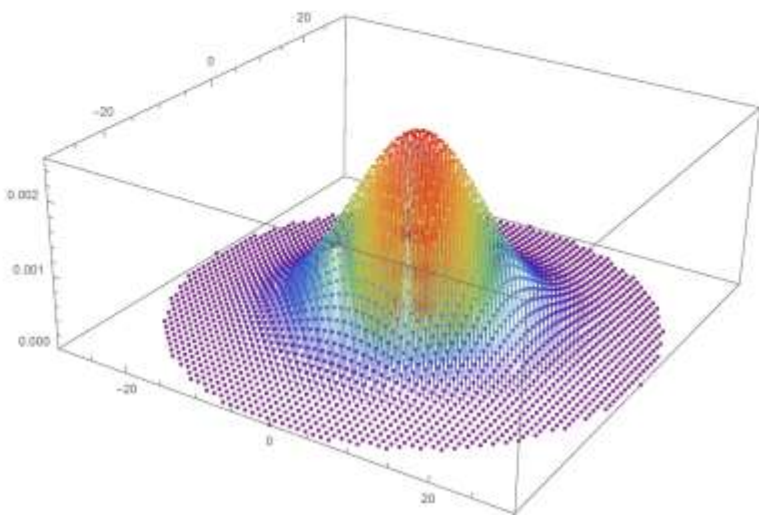
=

$$\frac{\mathbf{x}}{2} \sim \mathsf{Gauss}(s/2)$$

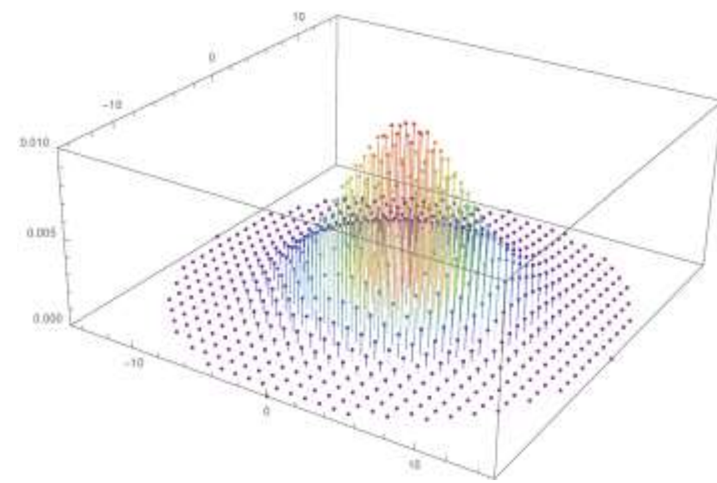


Discrete Gaussian Distribution

$\mathbf{y} \sim D_{\mathcal{L},s}$

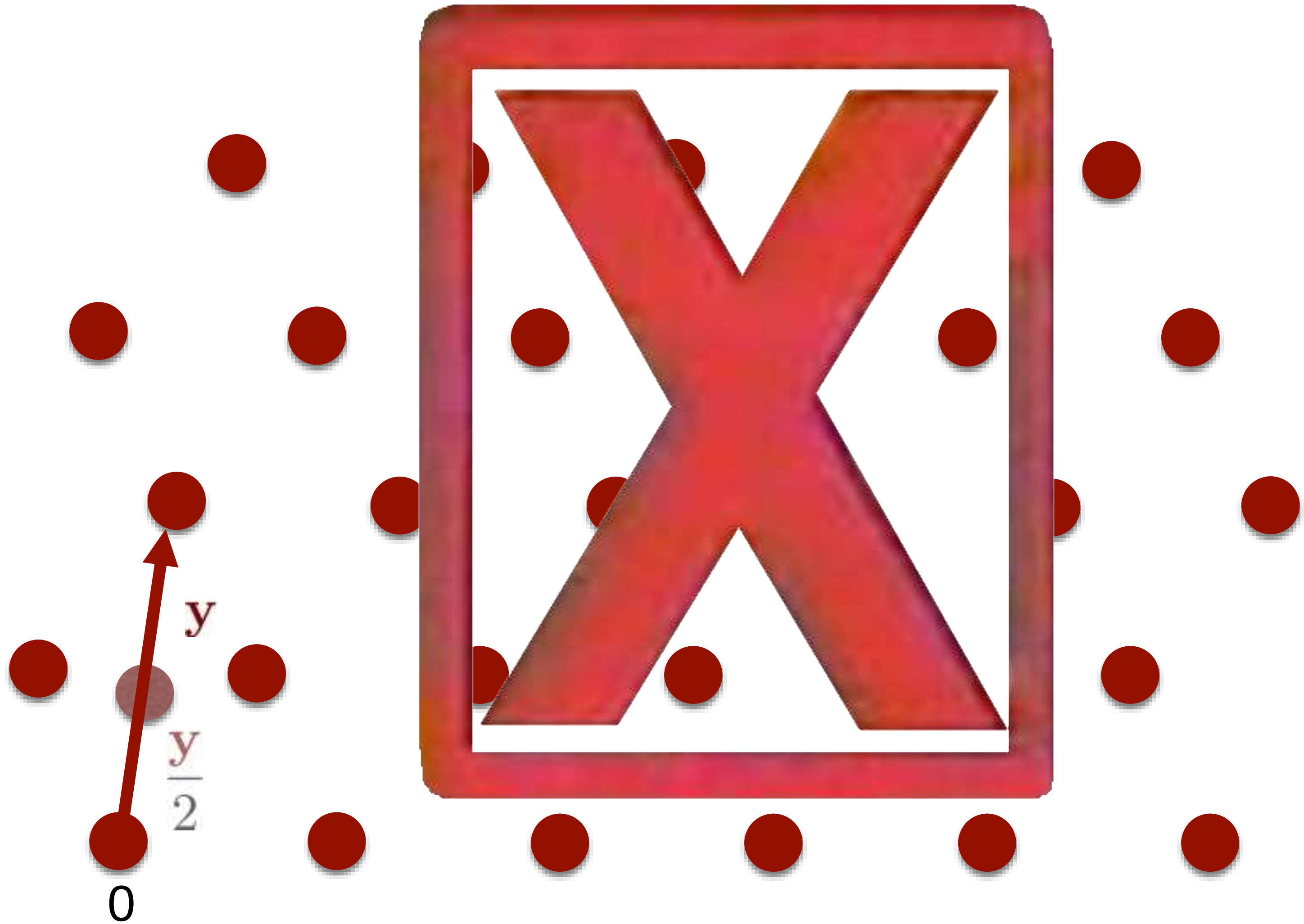


$\overset{?}{=}$



2

Discrete Gaussian Distribution



Converting Gaussian Vectors

What if we *condition on* the result being in the lattice?

$$\Pr_{\mathbf{y}\sim D_{\mathcal{L},s}}\left[\frac{\mathbf{y}}{2}=\mathbf{x}\;\middle|\;\frac{\mathbf{y}}{2}\in\mathcal{L}\right]\propto e^{-4||\mathbf{x}||^2/s^2}$$

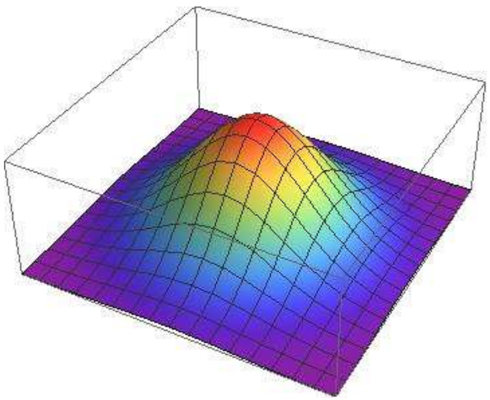
Progress!

Unfortunately, this requires us to throw out a lot of vectors.

We only keep one from every $\approx 2^n$ vectors each time we do this, leading to a very slow algorithm!

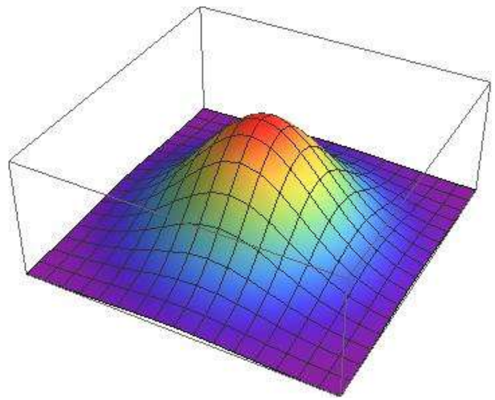
Converting Gaussian Vectors

$\mathbf{x}_1 \sim \text{Gauss}(s)$



+

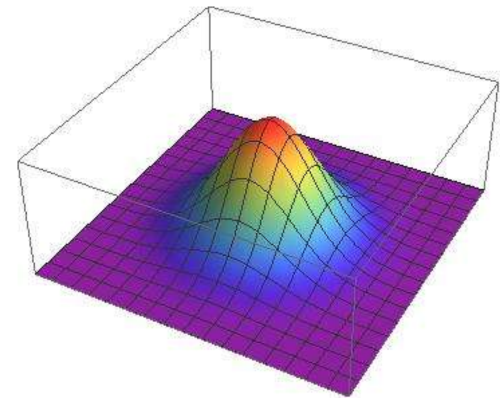
$\mathbf{x}_2 \sim \text{Gauss}(s)$



2

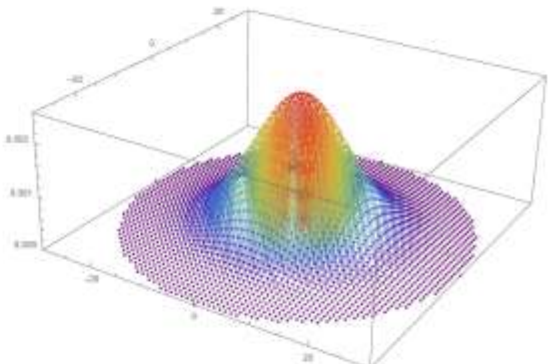
=

$\frac{\mathbf{x}_1 + \mathbf{x}_2}{2} \sim \text{Gauss}(s/\sqrt{2})$

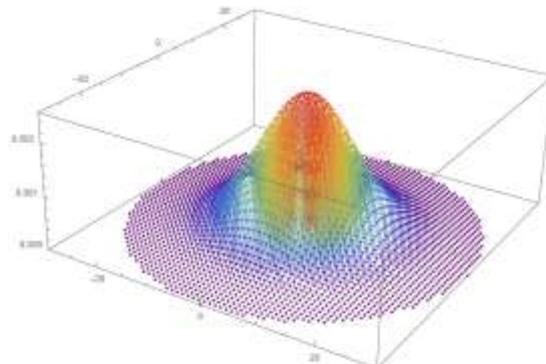


Converting Gaussian Vectors

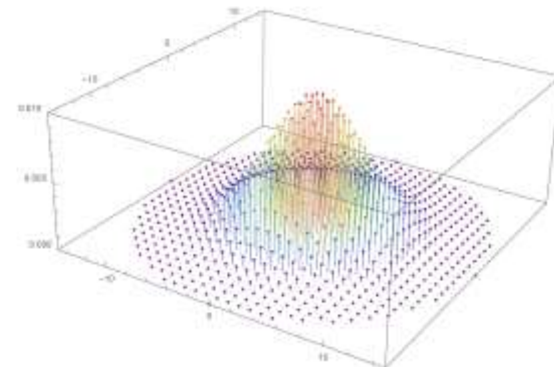
$\mathbf{y}_1 \sim D_{\mathcal{L},s}$ $\quad$ $\mathbf{y}_2 \sim D_{\mathcal{L},s}$ $\quad$ $\dfrac{\mathbf{y}_1 + \mathbf{y}_2}{2} \sim D_{\mathcal{L},s/\sqrt{2}}$



$+$

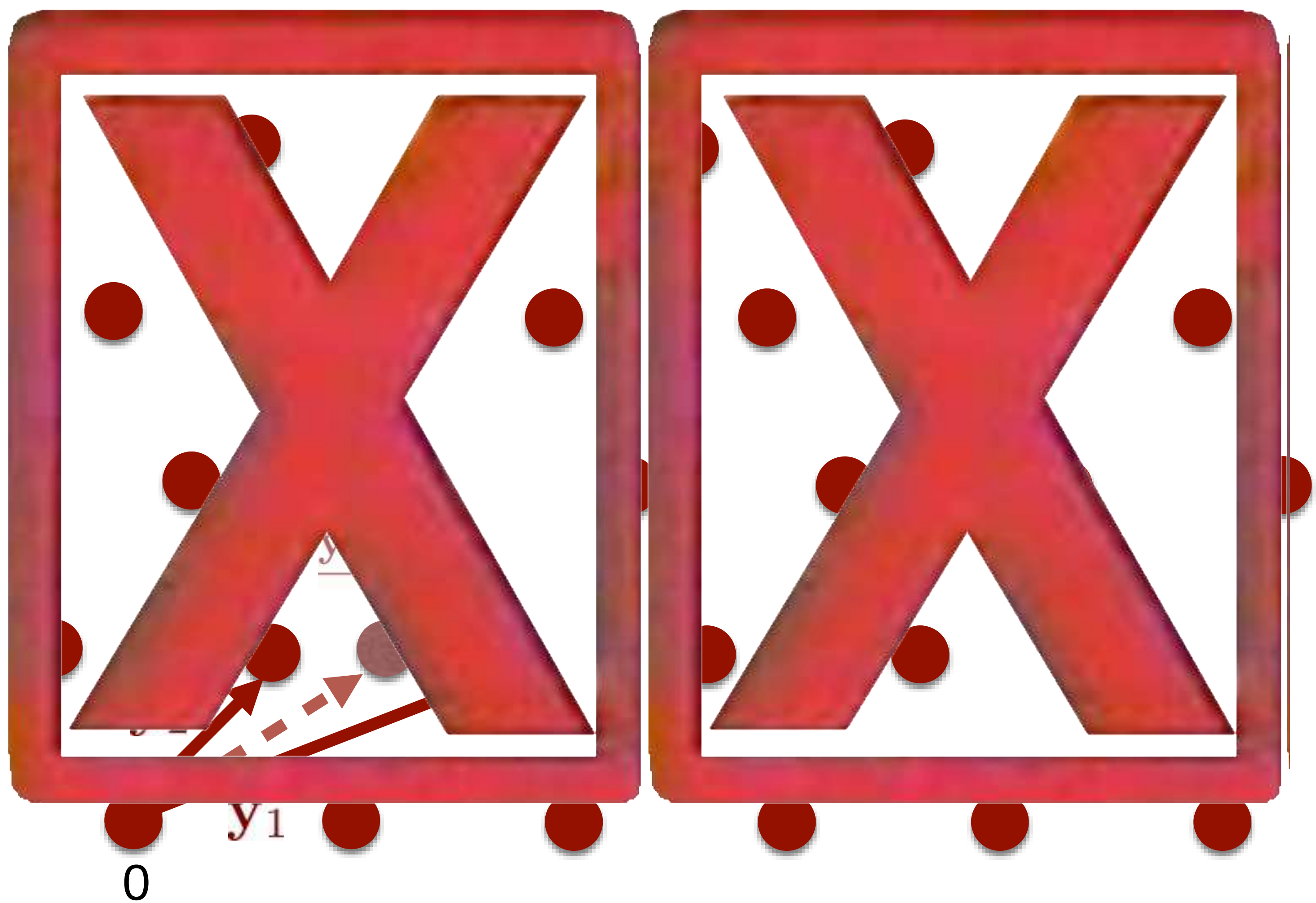


$\stackrel{?}{=}$



2

Converting Gaussian Vectors



Converting Gaussian Vectors

What about the average of two discrete Gaussian vectors *conditioned on* the result being in the lattice?

Converting Gaussian Vectors

When do we have $\frac{\mathbf{y}_1 + \mathbf{y}_2}{2} \in \mathcal{L}$?

$$\mathbf{y}_1 = a_{1,1}\mathbf{b}_1 + \cdots + a_{1,n}\mathbf{b}_n \qquad \mathbf{y}_2 = a_{2,1}\mathbf{b}_1 + \cdots + a_{2,n}\mathbf{b}_n$$

We have $(y_1 + y_2)/2 \in \mathcal{L}$ *if and only if* y_1, y_2 *are in the same* ***coset*** *of* $2\mathcal{L}$ *.*
(Note that there are 2^n *cosets)*

$$\frac{\mathbf{y}_1 + \mathbf{y}_2}{2} \in \mathcal{L} \iff a_{1,i} \equiv a_{2,i} \mod 2$$

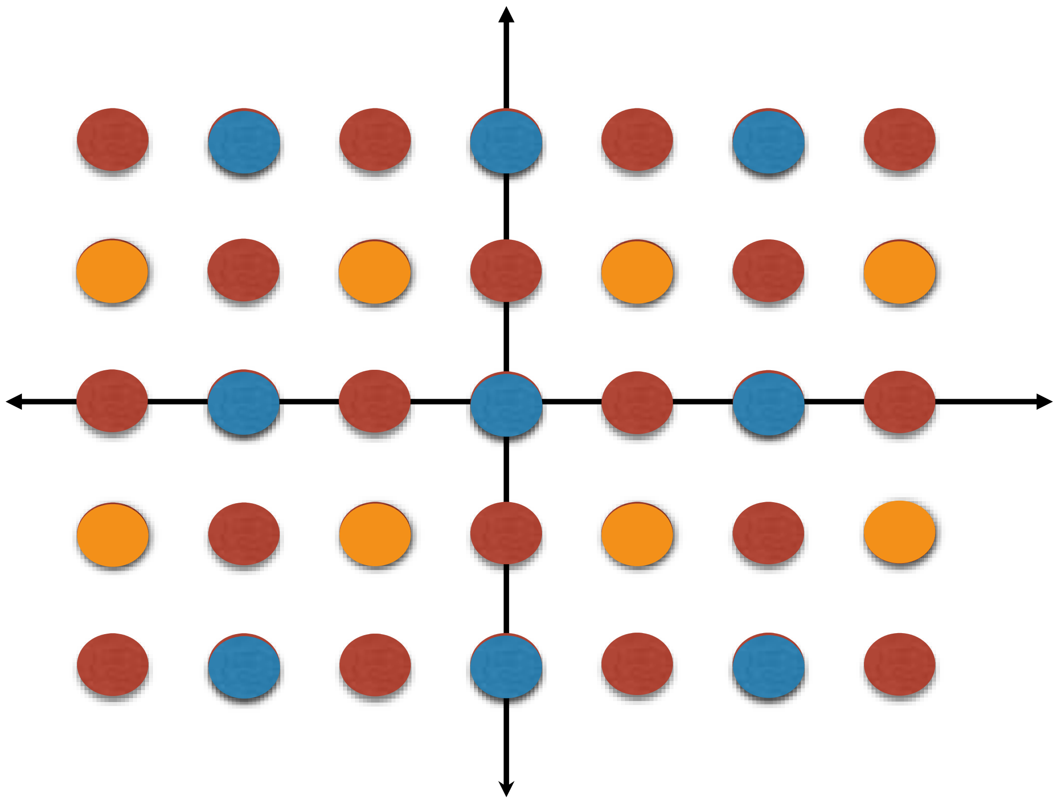
$$\iff \mathbf{y}_1 \equiv \mathbf{y}_2 \mod 2\mathcal{L}$$

Converting Gaussian Vectors

What about the average of two discrete Gaussian vectors *conditioned on* the result being in the lattice?

Converting Gaussian Vectors

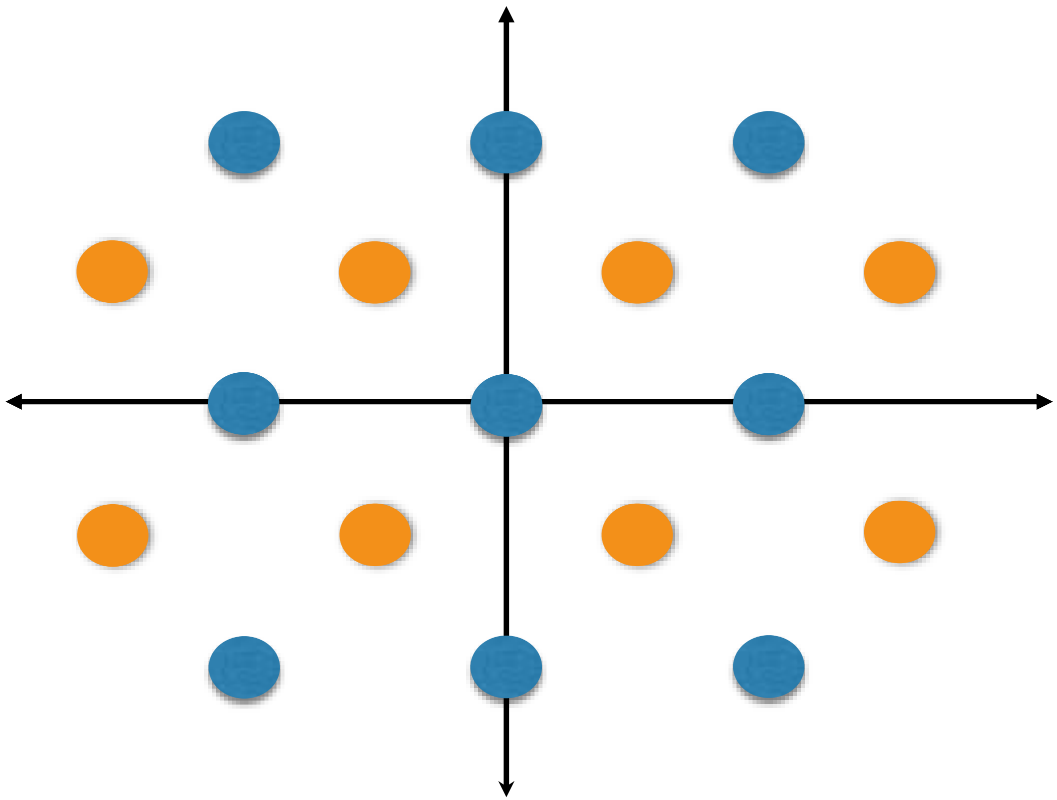
$$\mathcal{L}^\dagger = \{(y_1, y_2) \in \mathcal{L} \times \mathcal{L} : y_1 \equiv y_2 \pmod{2\mathcal{L}}\}$$



What about the average of two discrete Gaussian vectors *conditioned on* the result being in the lattice?

Converting Gaussian Vectors

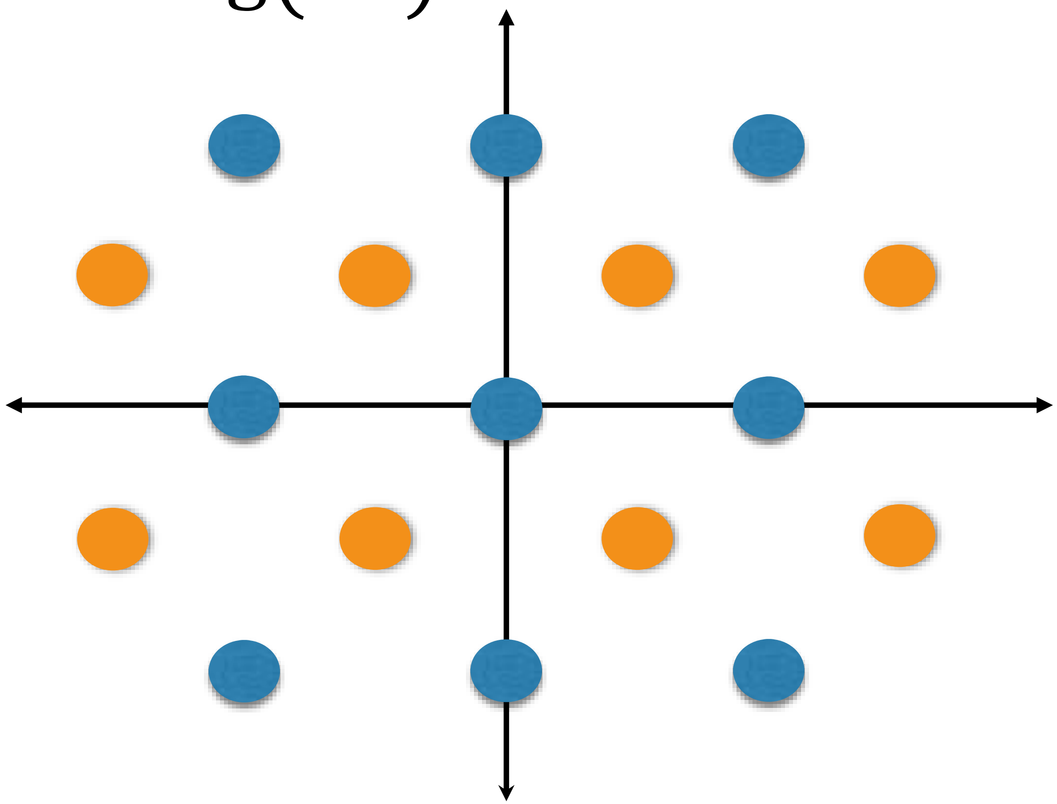
$$\text{avg}(y_1, y_2) = \left(\frac{y_1 + y_2}{2}, \frac{y_1 - y_2}{2}\right)$$



What about the average of two discrete Gaussian vectors *conditioned on* the result being in the lattice?

Converting Gaussian Vectors

$$\text{avg}(\mathcal{L}^\dagger) = \mathcal{L} \times \mathcal{L}$$



What about the average of two discrete Gaussian vectors *conditioned on* the result being in the lattice?

Converting Gaussian Vectors

$$\mathrm{avg}(\mathcal{L}^{\dagger}) = \mathcal{L} \times \mathcal{L}$$

If we sample $y_1, y_2 \sim D_{\mathcal{L},s}$,
then their average will be distributed as $D_{\mathcal{L},s/\sqrt{2}}$,
if we condition on the result being in the lattice.

$\|$ $\sqrt{2}$

$$(y_1, y_2) \sim D_{\mathcal{L}^{\dagger},s} \Rightarrow \mathrm{avg}(y_1, y_2) \sim D_{\mathcal{L}\times\mathcal{L},s/\sqrt{2}}$$

Progress!

$$D_{\mathcal{L},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}\|^2/s^2}$$

Stitching a Discrete Gaussian Together

$$\Pr_{y_1,y_2 \sim D_{\mathcal{L},s}}\left[\frac{y_1+y_2}{2} = y \mid \frac{y_1+y_2}{2} \in \mathcal{L}\right]$$

$$\propto \sum_{\boldsymbol{c} \in \mathcal{L} (\mathrm{mod}\ 2\mathcal{L})} \Pr\left[D_{\mathcal{L},s} \in \boldsymbol{c}\right]^2 \Pr_{y_1,y_2 \sim D_{2\mathcal{L}+\boldsymbol{c},s}}\left[\frac{y_1+y_2}{2} = y\right]$$

Generating a single $D_{\mathcal{L},s/\sqrt{2}}$ sample:

1. Sample $\boldsymbol{c} \in \mathcal{L}\ (mod\ 2\mathcal{L})$ with probability $\propto \Pr\left[\mathrm{D}_{\mathcal{L},s} \in \boldsymbol{c}\right]^2$.

2. Output $(Y_1 + Y_2)/2$ where $Y_1, Y_2 \sim D_{2\mathcal{L}+\boldsymbol{c},s}$.

Discrete Gaussian Combiner

Input: $Y_1, \ldots, Y_M$ iid $D_{\mathcal{L},s}$ samples ($M \approx 2^n$)

1. "Bucket" samples according to their coset $(\mathrm{mod}\ 2\mathcal{L})$.
2. Repeat many times:
   1. Sample coset $\boldsymbol{c}$ with probability $\propto \Pr\left[\mathrm{D}_{\mathcal{L},s} \in \boldsymbol{c}\right]^2$.
   2. Output $(Y_i + Y_j)/2$, for $Y_i, Y_j \in \boldsymbol{c}$.
   3. Remove Y_i, Y_j from list.

Don't have access to this distribution!

Rejection Sampling

Achieving $\propto \mathbf{Pr}\left[\boldsymbol{D}_{\mathcal{L},s} \in \boldsymbol{c}\right]^{2}$**:**

Same as trivial strategy!

First Pass:
Sample $\boldsymbol{c} \sim D_{\mathcal{L},s} \pmod{2\mathcal{L}}$.
Accept $\boldsymbol{c}$ with probability $\Pr[D_{\mathcal{L},s} \in \boldsymbol{c}]$ o/w reject.

Implementation:
Sample $Y_1 \sim D_{\mathcal{L},s}$ and let $\boldsymbol{c}$ be $Y_1 \pmod{2\mathcal{L}}$.
Sample $Y_2 \sim D_{\mathcal{L},s}$.
Output $\boldsymbol{c}$ if $Y_1 \equiv Y_2 \pmod{2\mathcal{L}}$.

Rejection Sampling

Achieving $\propto \mathbf{Pr}\left[\boldsymbol{D}_{\mathcal{L},\boldsymbol{s}} \in \boldsymbol{c}\right]^{\mathbf{2}}$**:**

Second Try:

Sample $\boldsymbol{c} \sim D_{\mathcal{L},s} \pmod{2\mathcal{L}}$.

Accept $\boldsymbol{c}$ with probability $\frac{\Pr\left[D_{\mathcal{L},s} \in \boldsymbol{c}\right]}{p_{\max}}$ o/w reject,

where

$$p_{\max} = \max_{\boldsymbol{b} \in \mathcal{L} \pmod{2\mathcal{L}}} \Pr[D_{\mathcal{L},s} \in \boldsymbol{b}]$$

Implementation: **???**

Discrete Gaussian Combiner

Input: $Y_1, \ldots, Y_M$ iid $D_{\mathcal{L},s}$ samples ($M \approx 2^n$)

Use first $M/6$ samples to estimate $p_{\max}$.

	1	…	$Mp_{\max}/3$
$\mathcal{L}(mod\ 2\mathcal{L})$ 2^n buckets	# samples in each bucket		
	First $1/p_{\max}$ samples	$\cdots$	Last $1/p_{\max}$ samples

Discrete Gaussian Combiner

Input: $Y_1, \ldots, Y_M$ iid $D_{\mathcal{L},s}$ samples ($M \approx 2^n$)

1. Compute $p_{\max}$ and bucket counts (previous slide).

2. For i ranging over last $M/6$ samples:
    1. Let $\boldsymbol{c} = Y_i\ (mod\ 2\mathcal{L})$.
    2. Find first unused bucket count $k_{\boldsymbol{c}}$ for coset $\boldsymbol{c}$.
    3. With probability $\min\ \{1, k_{\boldsymbol{c}}/n^{O(1)}\}$,

       output $(Y_i + Y_j)/2$

    where Y_j is any sample contributing to $k_{\boldsymbol{c}}$.

How Many Vectors Do We Get?

$M \;\coloneqq$ # input vectors

$$\text{\# output vectors} \approx M \cdot \frac{\sum_{\boldsymbol{c}} \Pr[D_{\mathcal{L},s} \in \boldsymbol{c}]^2}{\max_{\boldsymbol{b}} \Pr[D_{\mathcal{L},s} \in \boldsymbol{b}]}$$

Worst case bound: probability is at least $1/\sqrt{|\text{support}|}$.

Ma[illegible]op t[illegible] $/2$[illegible] afte[illegible]ingl[illegible]ep!

How Many Vectors Do We Get?

$$\#\text{ of output vectors } \approx M \cdot \frac{\sum_{\mathbf{c}} \Pr[D_{\mathcal{L},s} \in \mathbf{c}]^2}{\max_{\mathbf{c}} \Pr[D_{\mathcal{L},s} \in \mathbf{c}]}$$

$$\sum_{\mathbf{c}} \Pr[D_{\mathcal{L},s} \in \mathbf{c}]^2 = \frac{\rho_s(\mathcal{L}^{\dagger})}{\rho_s(\mathcal{L})^2}$$

$$= \frac{\rho_s(\sqrt{2}\mathcal{L} \times \sqrt{2}\mathcal{L})}{\rho_s(\mathcal{L})^2}$$

$$= \frac{\rho_{s/\sqrt{2}}(\mathcal{L})^2}{\rho_s(\mathcal{L})^2}$$

$$\rho_s(\mathcal{L}) \coloneqq \sum_{y \in \mathcal{L}} e^{-\|y/s\|^2}$$

How Many Vectors Do We Get?

$$\# \text{ of output vectors } \approx M \cdot \frac{\rho_{s/\sqrt{2}}(\mathcal{L})^2}{\rho_s(\mathcal{L})^2 \max_{\mathbf{c}} \Pr[D_{\mathcal{L},s} \in \mathbf{c}]}$$

$$\max_{\boldsymbol{c}} \rho_s(2\mathcal{L} + \mathbf{c}) = \rho_s(2\mathcal{L}) \longrightarrow \quad = M \cdot \frac{\rho_{s/\sqrt{2}}(\mathcal{L})^2}{\rho_s(\mathcal{L})\rho_s(2\mathcal{L})}$$

$$= M \cdot \frac{\rho_{s/\sqrt{2}}(\mathcal{L})^2}{\rho_s(\mathcal{L})\rho_{s/2}(\mathcal{L})}$$

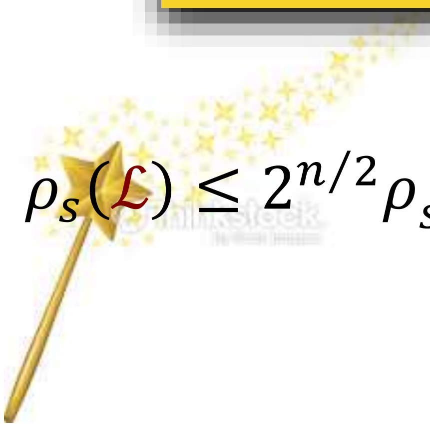
$$\rho_s(\mathcal{L}) := \sum_{y \in \mathcal{L}} e^{-\|y/s\|^2}$$

How Many Vectors Do We Get?

$$\#\ \text{of output vectors after } \ell \text{ steps} \ \approx M \cdot \prod_{i=0}^{\ell} \frac{\rho_{2^{-\frac{i+1}{2}}s}(\mathcal{L})^2}{\rho_{2^{-\frac{i}{2}}s}(\mathcal{L})\rho_{2^{-\frac{i+2}{2}}s}(\mathcal{L})}$$

Recall that we only need 1.38^n samples to solve SVP!

$$\ldots \frac{\ldots}{\rho_s(\mathcal{L})} \ldots \frac{\ldots(\mathcal{L})}{\rho_{2^{-\frac{\ell+2}{2}}s}(\mathcal{L})}$$



$\rho_s(\mathcal{L}) \le 2^{n/2}\rho_{s/\sqrt{2}}(\mathcal{L}) \longrightarrow \quad \ge M \cdot 2^{-n/2}$

Setting $M \approx 2^n$ gives $\#$ output vectors $\approx 2^{n/2}$

Key Estimates

Poisson summation formula: "nice" function f

$$\sum_{y\in\mathcal{L}} f(y+\mathbf{t}) = \frac{1}{\det(\mathcal{L})} \sum_{x\in\mathcal{L}^*} \hat{f}(x)\, e^{2\pi i\langle x,\mathbf{t}\rangle}$$

Plug in $e^{-\pi\|x/s\|^2}$:

$$\rho_s(\mathcal{L}+\mathbf{t}) = \frac{s^n}{\det(\mathcal{L})} \sum_{x\in\mathcal{L}^*} e^{-\pi\,\|sx\|^2}\, e^{2\pi i\langle x,\mathbf{t}\rangle}$$

$$\rho_s(\mathcal{L}) = \frac{s^n}{\det(\mathcal{L})}\, \rho_{1/s}(\mathcal{L}^*)$$

Key Estimates

$$\rho_s(\mathcal{L} + \mathbf{t}) = \frac{s^n}{\det(\mathcal{L})} \sum_{x \in \mathcal{L}^*} e^{-\pi \|sx\|^2} e^{2\pi i \langle x, \mathbf{t} \rangle}$$

$$\rho_s(\mathcal{L}) = \frac{s^n}{\det(\mathcal{L})} \rho_{1/s}(\mathcal{L}^*)$$

Corollary 1: $\max_{\mathbf{t}} \rho_s(\mathcal{L} + \mathbf{t}) = \rho_s(\mathcal{L})$

Corollary 2: $\rho_{\alpha s}(\mathcal{L}) \leq \alpha^n \rho_s(\mathcal{L})$ for $\alpha \geq 1$.

Final Algorithm

SVPSolver$(\mathcal{L})$

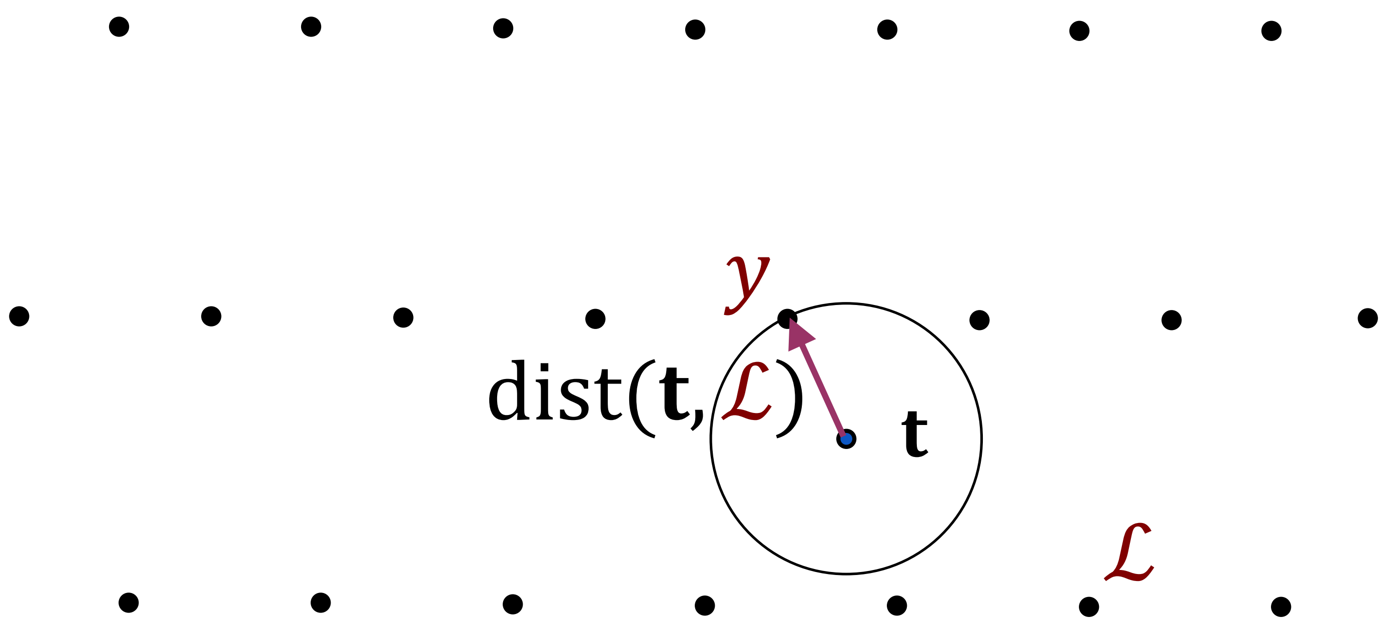
1. Use GPV to get $\approx 2^n$ samples from $D_{\mathcal{L},s}$ with $s \gg \lambda_1(\mathcal{L})$.
2. Run the ("squaring") discrete Gaussian combiner on the result repeatedly.
3. Output $\approx 2^{n/2}$ samples from $D_{\mathcal{L},s}$ with $s \approx \lambda_1(\mathcal{L})/\sqrt{n}$.
4. We can then simply output a shortest non-zero vector from our samples.

Act II: The Closest Vector Problem

Closest Vector Problem (CVP)

Given: Lattice basis $B \in \mathbb{Q}^{n \times n}$, target $\mathbf{t} \in \mathbb{Q}^n$.

Goal: Compute $y \in \mathcal{L}(B)$ minimizing $||\mathbf{t} - y||$.



Closest Vector Problem (CVP)

CVP seems to be the harder problem:

there is a dimension preserving reduction from **SVP** to **CVP** [GMSS99].

Algorithms for CVP

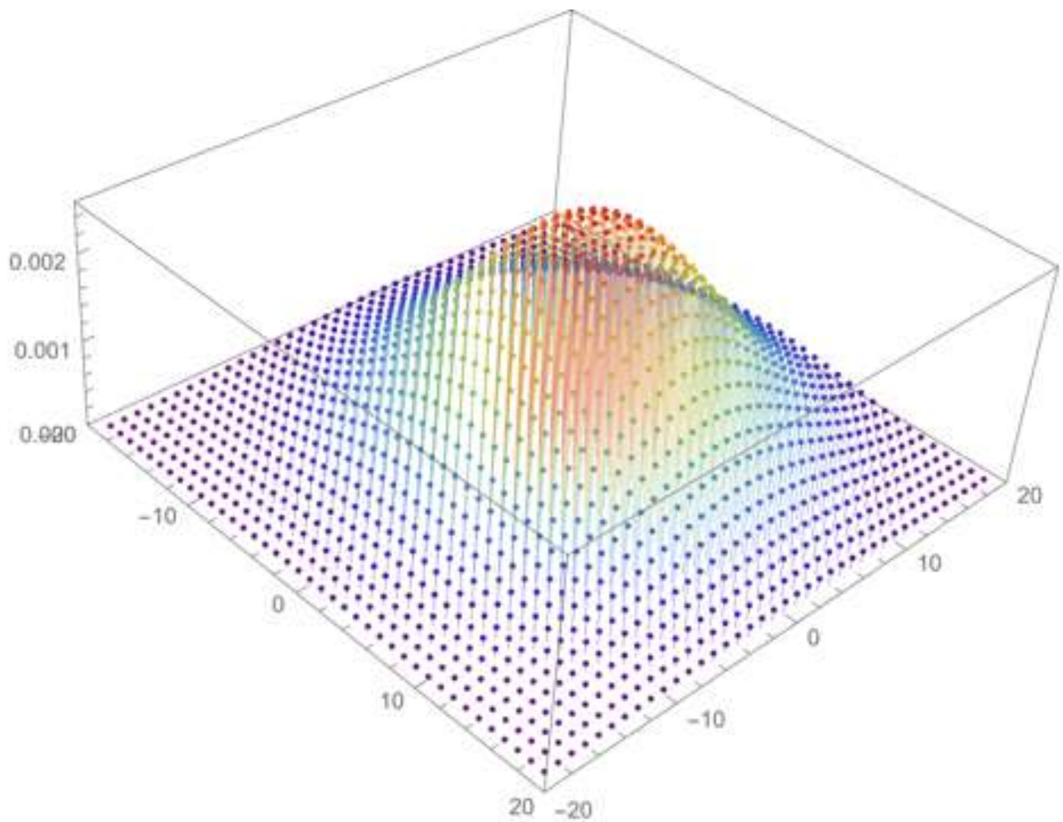
	Time	CVP?	Deterministic?
[Kan86,HS07,MW15] (Enumeration)	$n^{O(n)}$	Yes	Yes
[AKS02, BN09, HPS11, …] (Sieving)	$2^{O(n)}$	Approximate	No
[MV10b] (Voronoi cell)	$2^{2n+o(n)}$	Yes	Yes
[ADRS15] (Discrete Gaussian)	$2^{n+o(n)}$	Approximate	No
[ADS15]	$2^{n+o(n)}$	Yes	No

Disclaimer

The algorithm is quite complicated, so the following is a over-simplified high level sketch.

The Discrete Gaussian Distribution

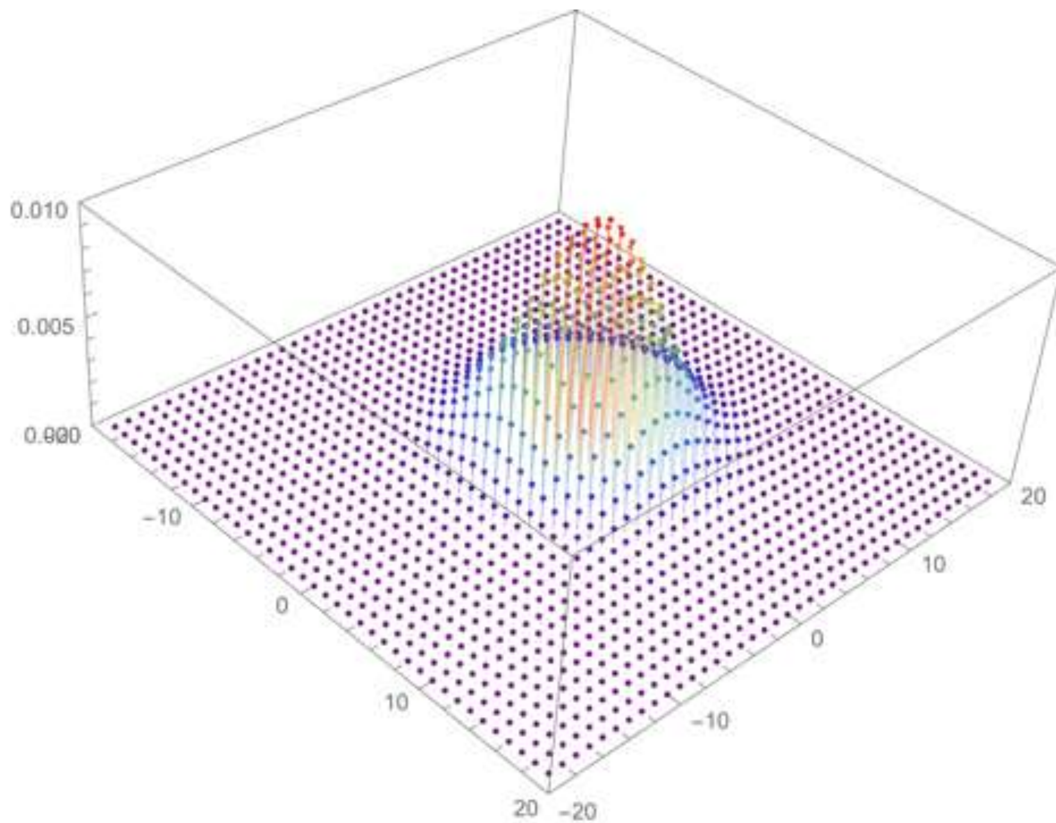
$$D_{\mathcal{L},\mathbf{t},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}-\mathbf{t}\|^2/s^2}$$



$s = 20$ $\mathcal{L} := \mathbb{Z}^2$ $\mathbf{t} := (0, 5/2)$

The Discrete Gaussian Distribution

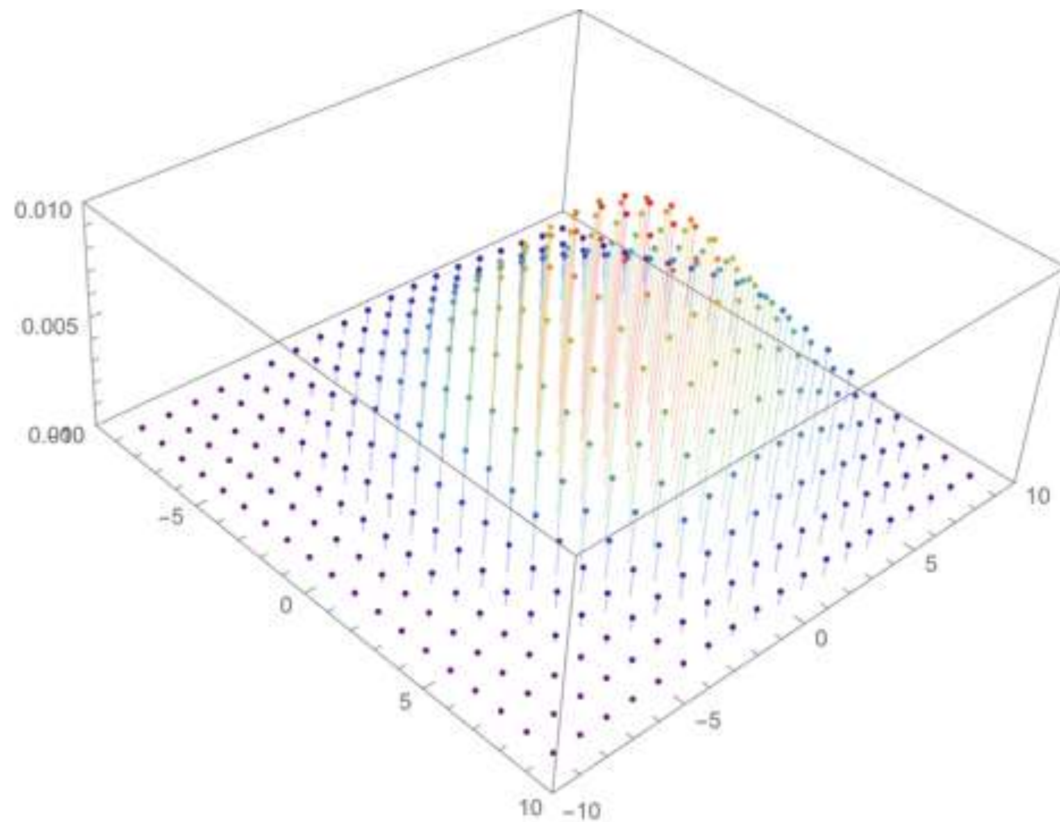
$$D_{\mathcal{L},\mathbf{t},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}-\mathbf{t}\|^2/s^2}$$



$s = 10$ $\mathcal{L} := \mathbb{Z}^2$ $\mathbf{t} := (0, 5/2)$

The Discrete Gaussian Distribution

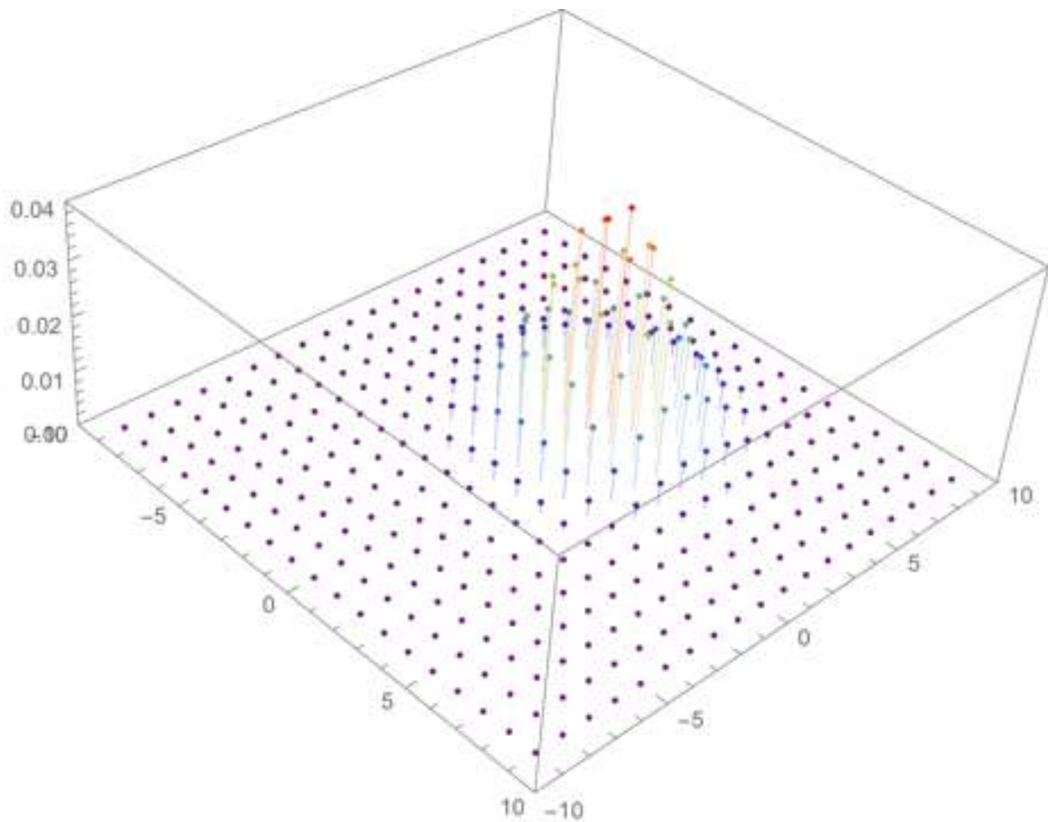
$$D_{\mathcal{L},\mathbf{t},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}-\mathbf{t}\|^2/s^2}$$



$s = 10$ $\mathcal{L} := \mathbb{Z}^2$ $\mathbf{t} := (0, 5/2)$

The Discrete Gaussian Distribution

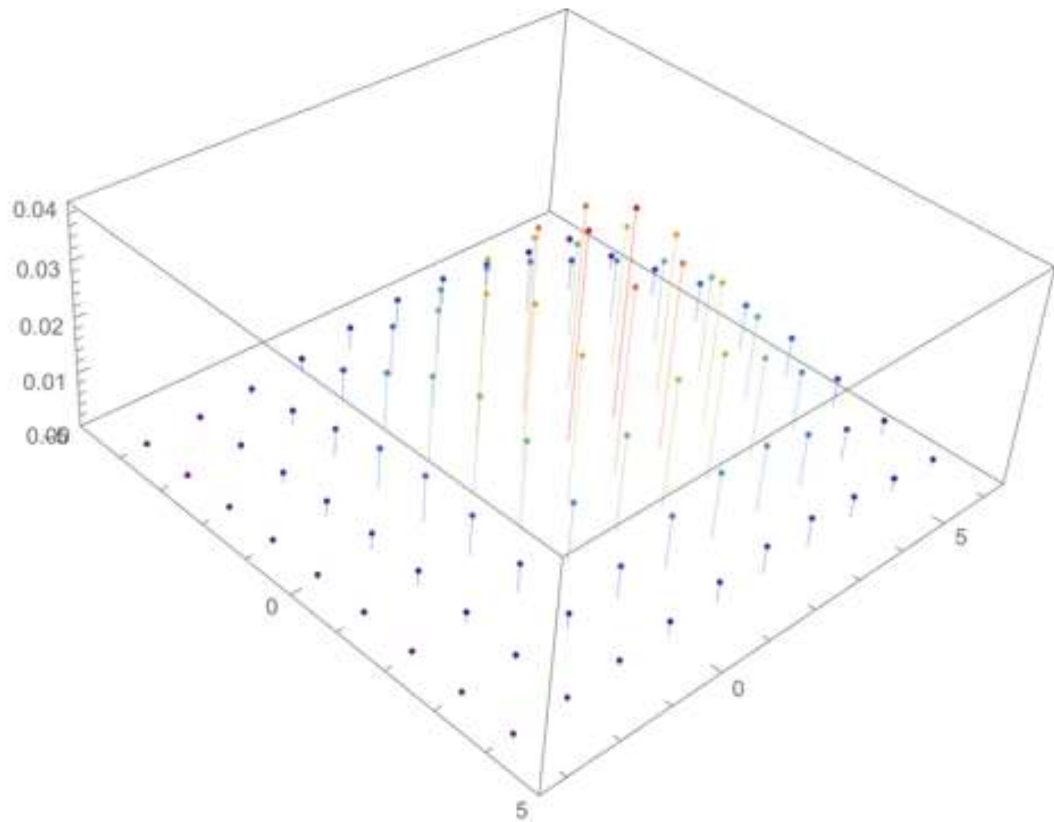
$$D_{\mathcal{L},\mathbf{t},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}-\mathbf{t}\|^2/s^2}$$



$s = 5$ $\mathcal{L} := \mathbb{Z}^2$ $\mathbf{t} := (0, 5/2)$

The Discrete Gaussian Distribution

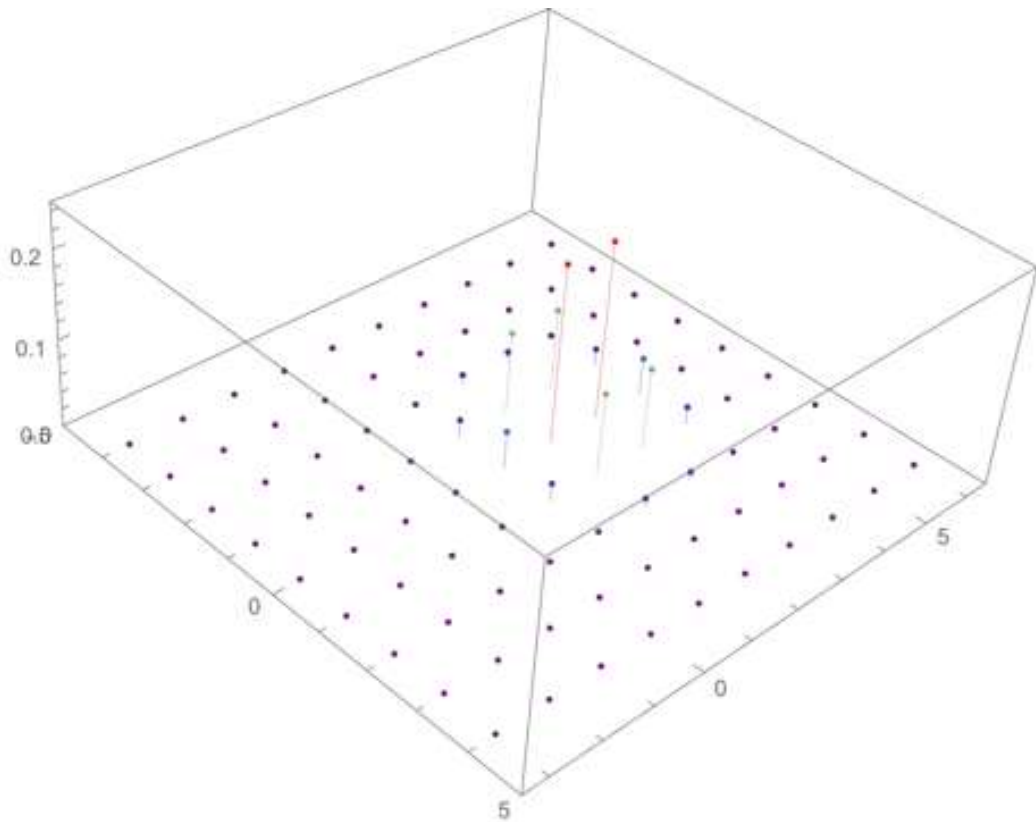
$$D_{\mathcal{L},\mathbf{t},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}-\mathbf{t}\|^2/s^2}$$



$s = 5$ $\mathcal{L} := \mathbb{Z}^2$ $\mathbf{t} := (0, 5/2)$

The Discrete Gaussian Distribution

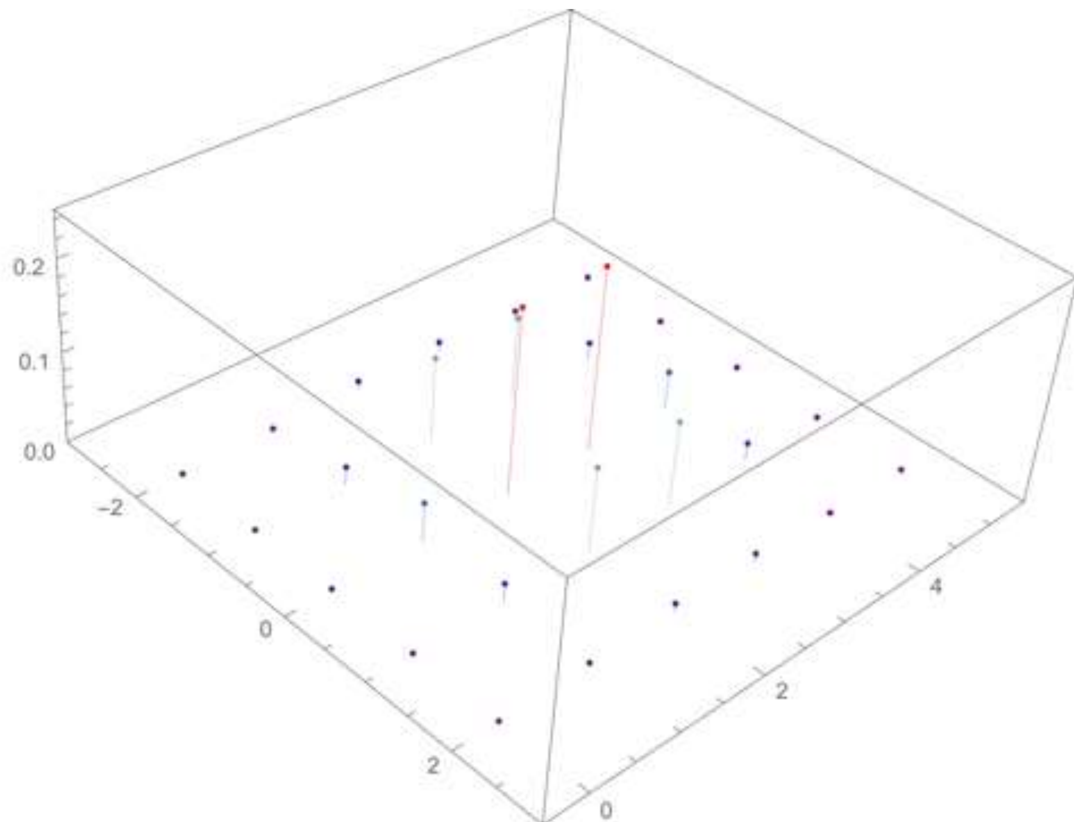
$$D_{\mathcal{L},\mathbf{t},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}-\mathbf{t}\|^2/s^2}$$



$s = 2$ $\mathcal{L} := \mathbb{Z}^2$ $\mathbf{t} := (0, 5/2)$

The Discrete Gaussian Distribution

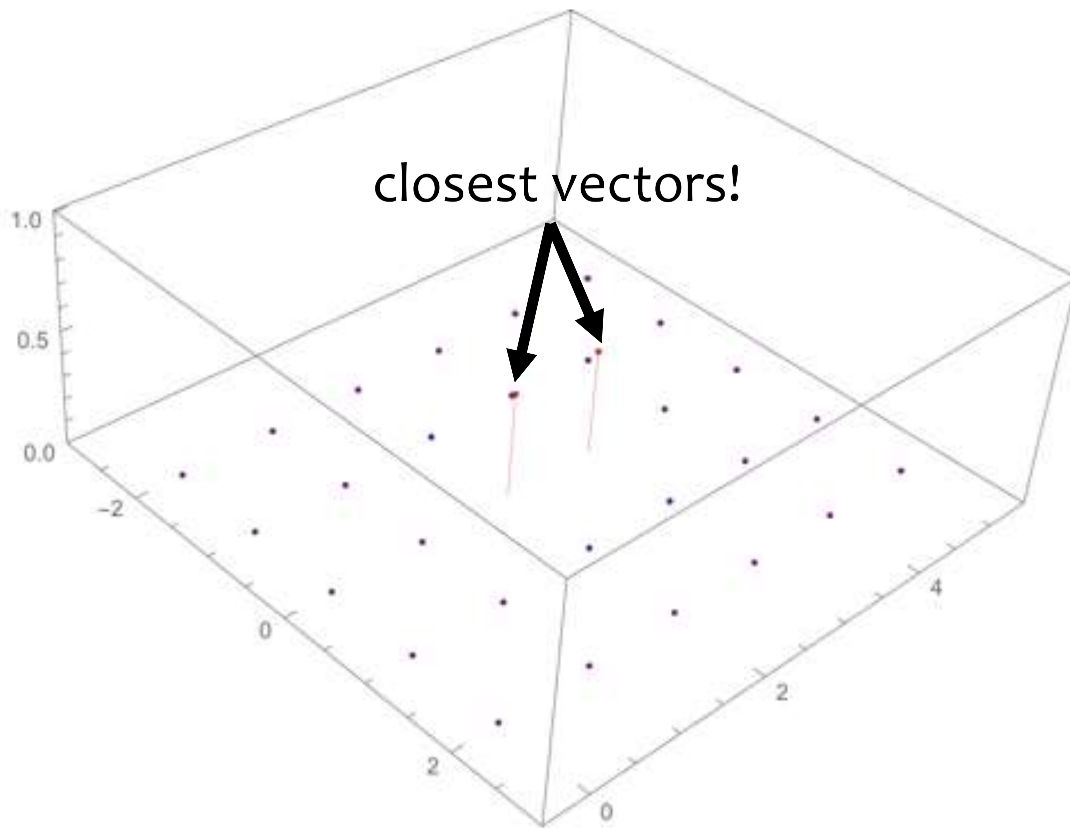
$$D_{\mathcal{L},\mathbf{t},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}-\mathbf{t}\|^2/s^2}$$



$s = 2$ $\qquad$ $\mathcal{L} := \mathbb{Z}^2$ $\qquad$ $\mathbf{t} := (0, 5/2)$

The Discrete Gaussian Distribution

$$D_{\mathcal{L},\mathbf{t},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}-\mathbf{t}\|^2/s^2}$$

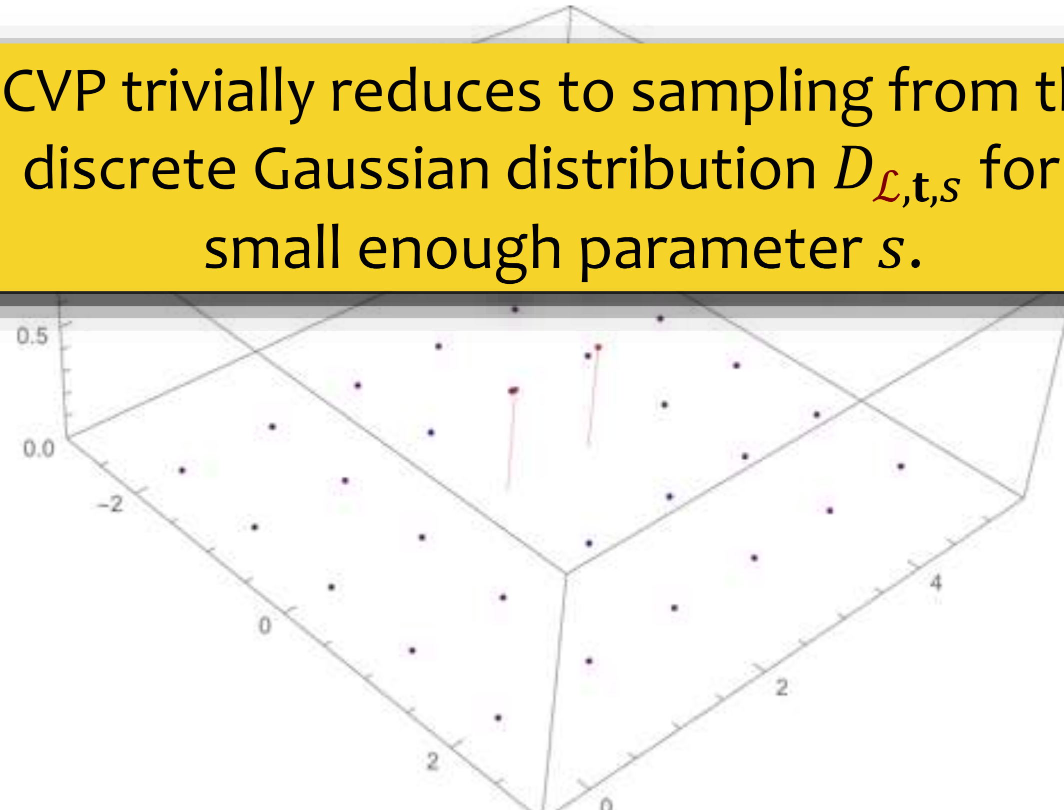


$s = 1$ $\mathcal{L} := \mathbb{Z}^2$ $\mathbf{t} := (0, 5/2)$

The Discrete Gaussian Distribution

$$D_{\mathcal{L},\mathbf{t},s} := \Pr[\mathbf{y}] \propto e^{-\|\mathbf{y}-\mathbf{t}\|^2/s^2}$$

CVP trivially reduces to sampling from the discrete Gaussian distribution $D_{\mathcal{L},\mathbf{t},s}$ for a small enough parameter s.



$s = 1$ $\mathcal{L} := \mathbb{Z}^2$ $\mathbf{t} := (0, 5/2)$

"Rotation" Identity Generalizes

$$\Pr_{\mathbf{y}_1,\mathbf{y}_2 \sim D_{\mathcal{L},s}}\left[\frac{\mathbf{y}_1+\mathbf{y}_2}{2} = \mathbf{y} \;\middle|\; \frac{\mathbf{y}_1+\mathbf{y}_2}{2} \in \mathcal{L}\right] = \Pr_{\mathbf{x} \sim D_{\mathcal{L},s/\sqrt{2}}}[\mathbf{x} = \mathbf{y}]$$

Great! So, we just need to run the squaring combiner and we're done! Right!?

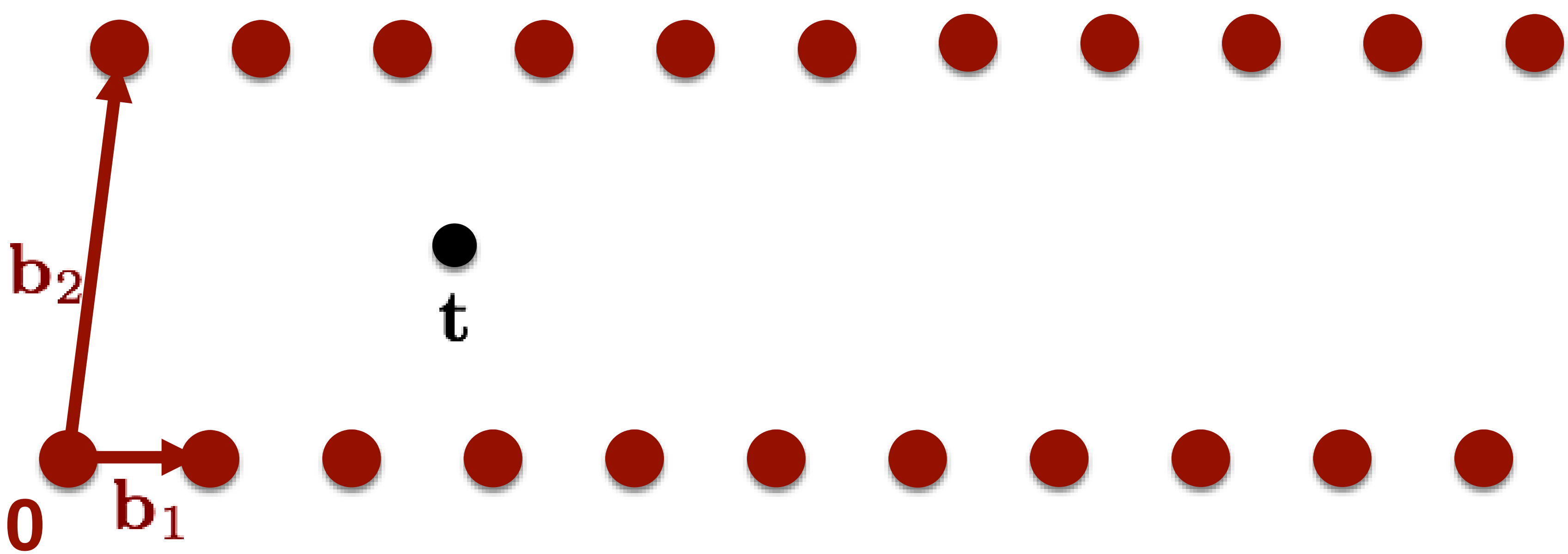
$$\Pr_{\mathbf{y}_1,\mathbf{y}_2 \sim D_{\mathcal{L},\mathbf{t},s}}\left[\frac{\mathbf{y}_1+\mathbf{y}_2}{2} = \mathbf{y} \;\middle|\; \frac{\mathbf{y}_1+\mathbf{y}_2}{2} \in \mathcal{L}\right] = \Pr_{\mathbf{x} \sim D_{\mathcal{L},\mathbf{t},s/\sqrt{2}}}[\mathbf{x} = \mathbf{y}]$$

Initialization Issues

- The [GPV08] sampler *does* work for sampling shifted $D_{\mathcal{L},\mathbf{t},s}$, but giv

Even if apply the combiner n times, we can only sample at $s \approx 2^{-n}\mathrm{dist}(\mathbf{t},\mathcal{L})$.

- When
  effectiv
- When $\mathbf{t} \neq 0$, we may not be able to do this.
- So, we must initialize with $s \gtrsim \mathrm{dist}(\mathbf{t},\mathcal{L})$.



Combiner Loss Factor

Going from $s \to s/\sqrt{2}$:

Cente

No obvious "magical cancelation".

General $\mathbf{t}$: $$\frac{\rho_{s/\sqrt{2}}(\mathcal{L})\rho_{s/\sqrt{2}}(\mathcal{L}-\mathbf{t})}{\rho_s(\mathcal{L})\max_{\boldsymbol{c}\in\mathcal{L}/2\mathcal{L}}\rho_s(\boldsymbol{c}-\mathbf{t})}$$

Combiner Loss Factor

Theorem: Combiner loss going from

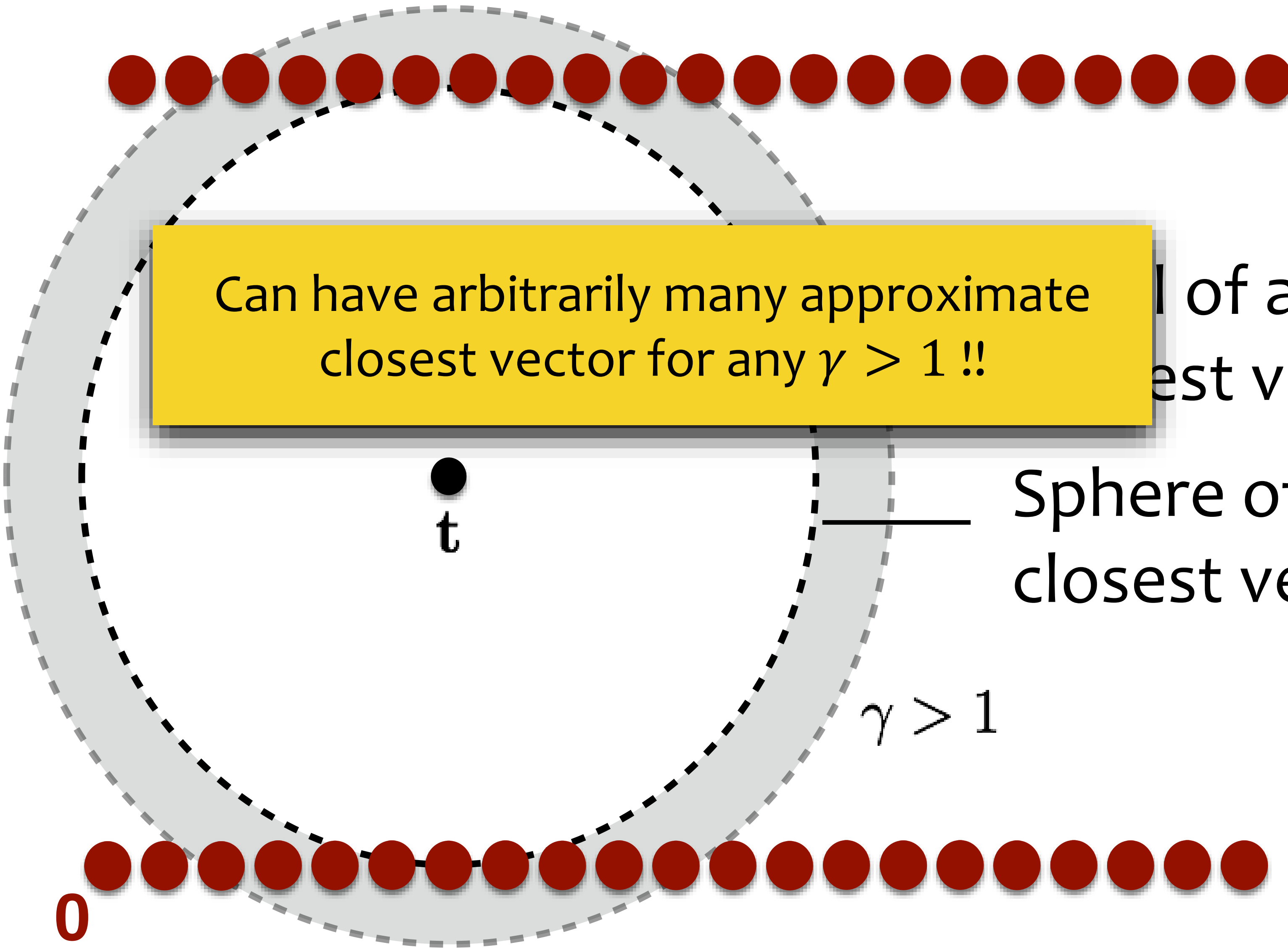
$$s \to s_k := s2^{-k/2}$$

is no worse than

$$\frac{2^{-n}}{\max_{\boldsymbol{c} \in \mathcal{L}/2\mathcal{L}} \Pr[D_{\mathcal{L},\mathbf{t},s_k} \in \boldsymbol{c}]}.$$

If we start with $2^{n+o(n)}$ samples, we always "see" the heaviest coset at each stage.

Exact vs Approximate CVP



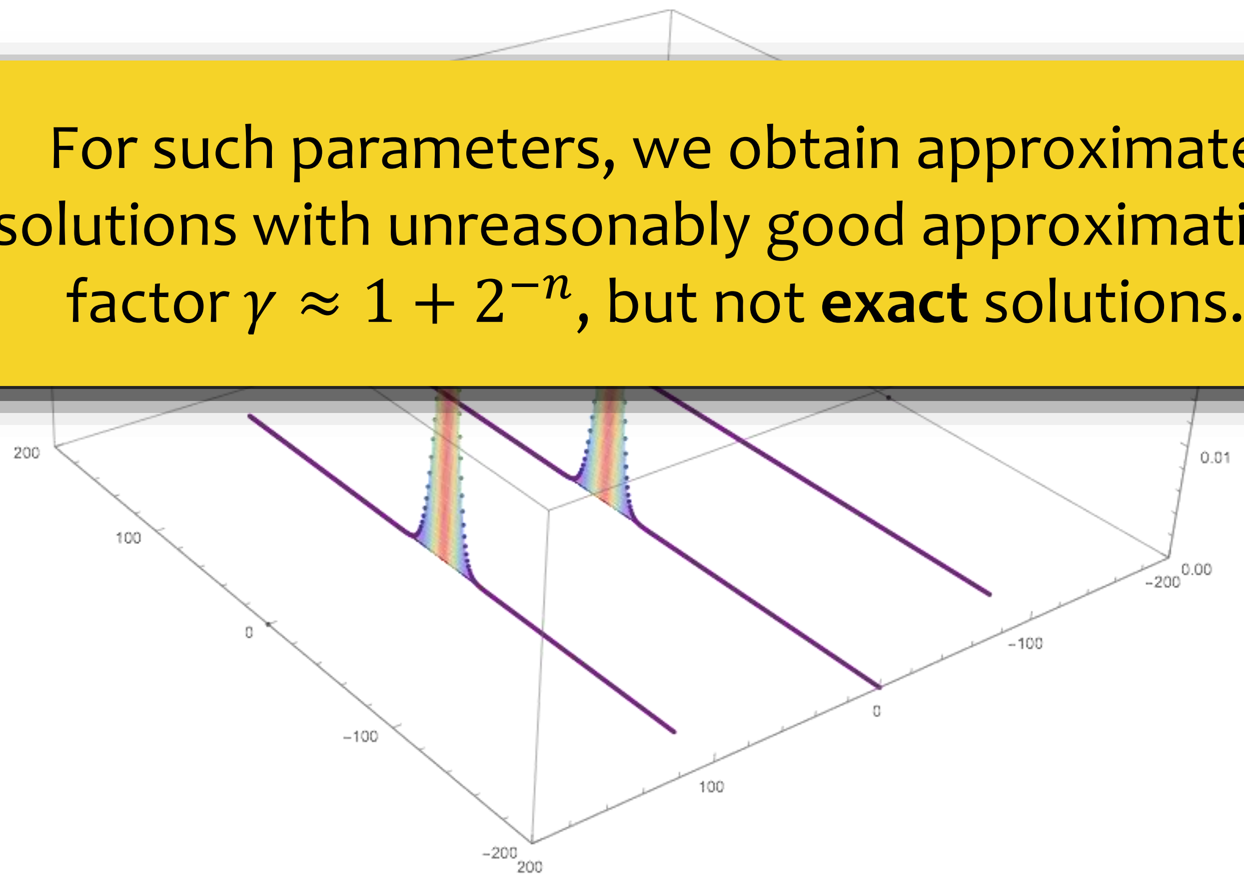
Can have arbitrarily many approximate closest vector for any $\gamma > 1$!!

l of approx
est vectors

Sphere of
closest vectors

We Need Small Parameters

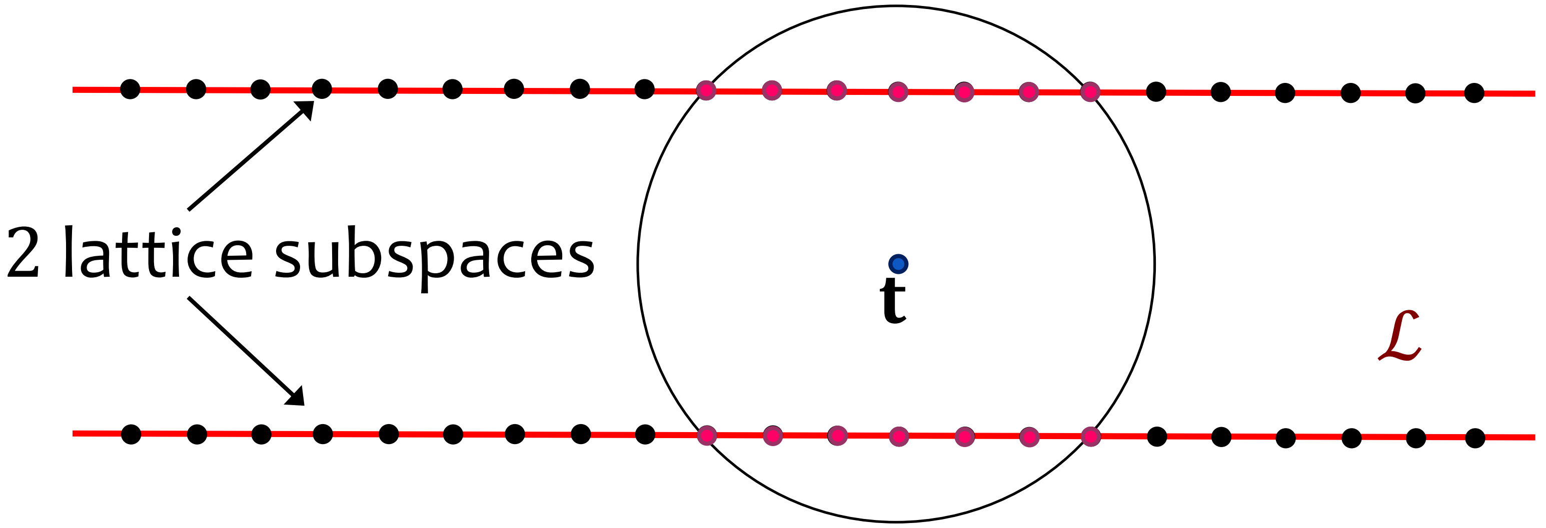
The reduction from CVP to DGS needs $s \ll \lambda_1(\mathcal{L})$, but we can only handle $s \approx 2^{-n} \cdot \mathrm{dist}(\mathbf{t}, \mathcal{L})$.



For such parameters, we obtain approximate solutions with unreasonably good approximation factor $\gamma \approx 1 + 2^{-n}$, but not **exact** solutions.

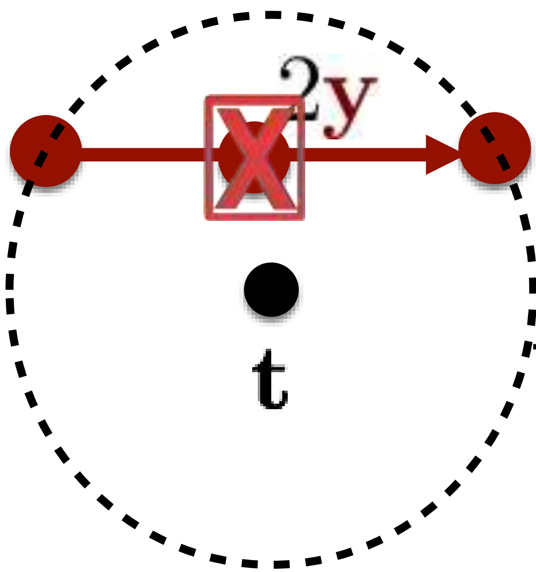
Hope for exact CVP

To apply recursion, need to identify them and show that there are not too many.
$2^{n+o(n)}$ time := at most 2 sub-problems per dimension!



Clusters

Claim: There are at most 2^n exact closest vectors.

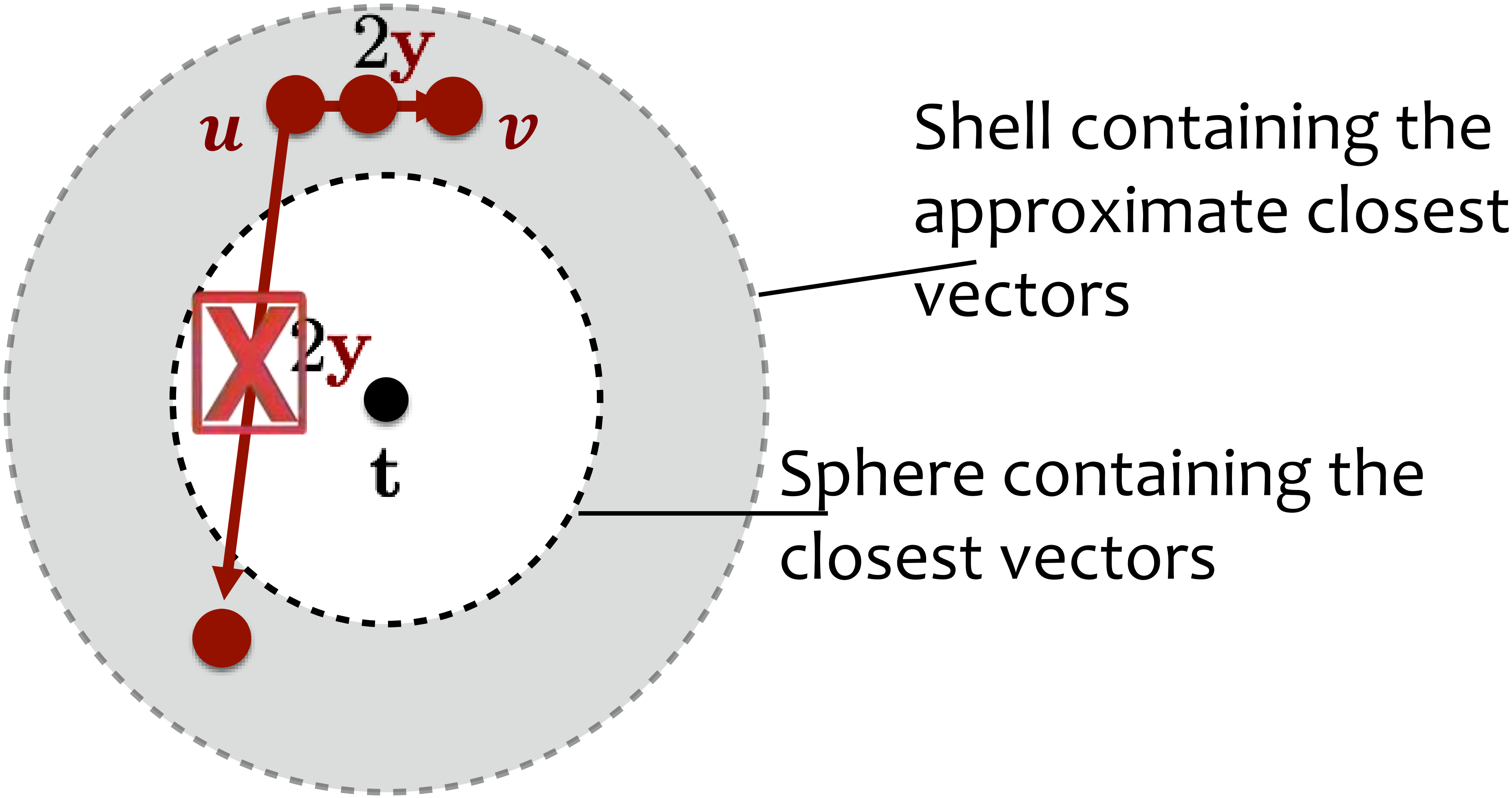


Must lie in different cosets of $\mathcal{L}/2\mathcal{L}$.

Sphere containing the closest vectors

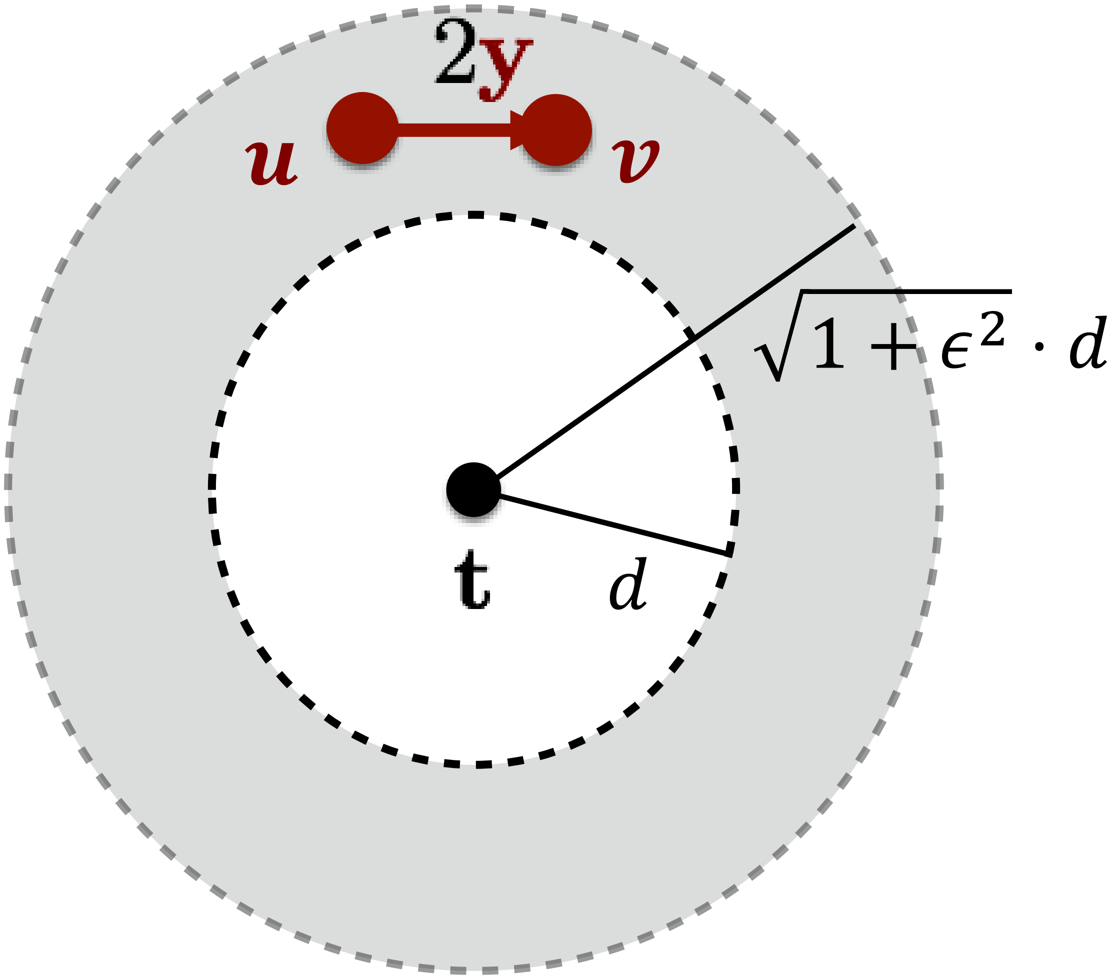
Clusters

Claim: The approximate closest vectors are contained in 2^n ``clusters'' of small diameter.



Clusters

Claim: $\sqrt{1+\epsilon^2}$ approx. CVP sols u and v.
$v - u \in 2\mathcal{L}$ implies $\|v - u\| \le 2\epsilon \cdot \mathrm{dist}(\mathbf{t}, \mathcal{L})$.



Clusters

Claim: $\sqrt{1+\epsilon^2}$ approx. CVP sols u and v.
$v - u \in 2\mathcal{L}$ implies $\|v - u\| \leq 2\epsilon \cdot \mathrm{dist}(\mathbf{t}, \mathcal{L})$.

$$\begin{aligned}
\|v-u\|^2 &= 2\|v-\mathbf{t}\|^2 + 2\|u-\mathbf{t}\|^2 \\
&\qquad -4\|(v+u)/2-\mathbf{t}\|^2 \\
&\leq 4(1+\epsilon^2)\cdot \mathrm{dist}(\mathbf{t},\mathcal{L})^2 \\
&\qquad -4\cdot \mathrm{dist}(\mathbf{t},\mathcal{L})^2 \\
&= 4\epsilon^2 \cdot \mathrm{dist}(\mathbf{t},\mathcal{L})^2
\end{aligned}$$

Taking advantage of clusters

"nearly orthogonal" basis $b_1, \dots, b_n$ of $\mathcal{L}$
(lengths in approx. non-decreasing order)

$1 + 2^{-n}$ approx CVP sols $y_1, \dots, y_N$ for $\mathbf{t}$.

$$y_j = \sum_i a_{i,j} b_i \ \forall j$$

Theorem: $\exists k$ such that last k coefficients

$$\{(a_{n-k+1,j}, \dots, a_{n,j}) : j \in [N]\}$$

come from set of size $\approx 2^k$. Recurse on these!

Taking advantage of clusters

Assume: **orthogonal** lattice $\mathcal{L}$

$$\mathcal{L} = \{(x_1 b_1, \ldots, x_n b_n) : x \in \mathbb{Z}^n\}$$
$$(0 \leq b_1 \leq \cdots \leq b_n)$$

For $\epsilon = 2^{-n}$, **all** coordinates are fixed by parity unless there are **exponential** gaps in basis vector lengths. But such gaps can exist….

Claim: If $y_r - y_s \in 2\mathcal{L}$ and $b_{n-k+1} > \sqrt{n}\epsilon b_n$

then $\left(a_{n-k+1,r}, \ldots, a_{n,r}\right) = (a_{n-k+1,s}, \ldots, a_{n,s})$

Taking advantage of clusters

Claim: If $y_r - y_s \in 2\mathcal{L}$ and $b_{n-k+1} \geq \sqrt{n}\epsilon b_n$

1. $\mathrm{dist}(\mathbf{t}, \mathcal{L}) \leq \frac{1}{2}\sqrt{\sum_i b_i^2} \leq \frac{\sqrt{n}}{2} b_n$

This shows we have at most 2^n clusters each of which is $n-k$ dimensional, but we need 2^k clusters!!!

$-k+1$

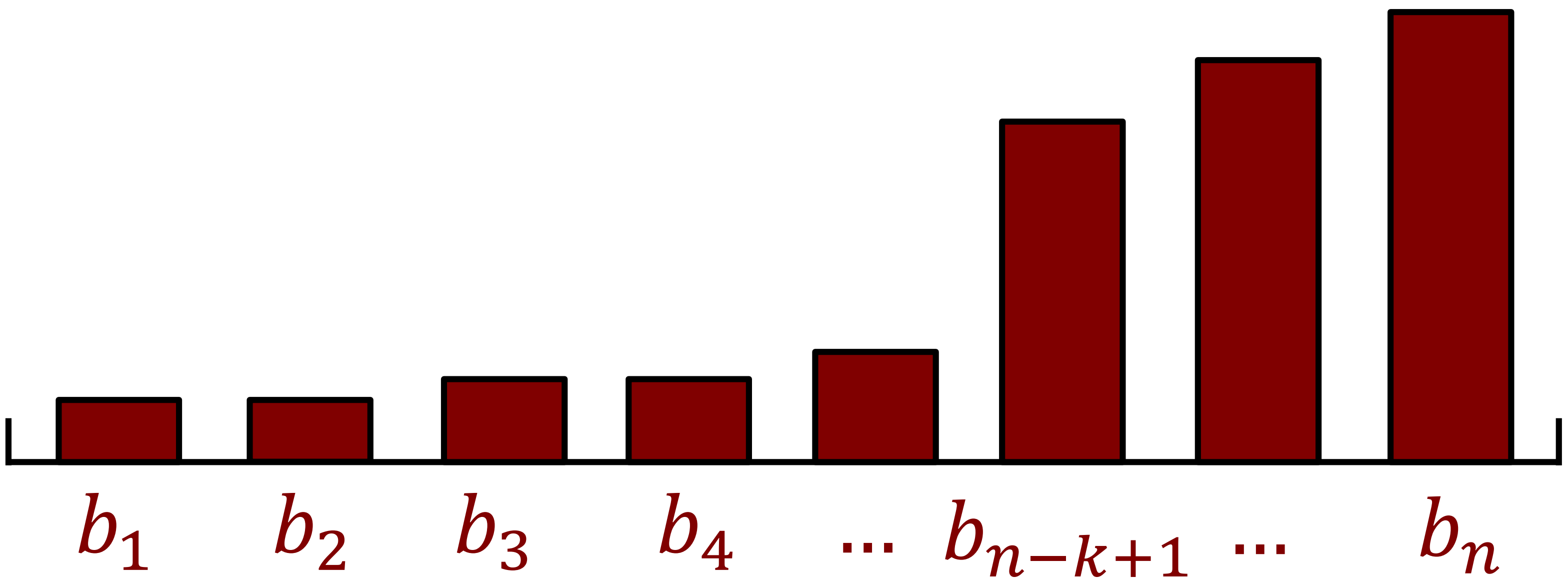
If y_r, y_s differ on any coordinate

$$i \in \{n-k+1, \dots, n\}$$

their difference would have norm at least b_{n-k+1}.

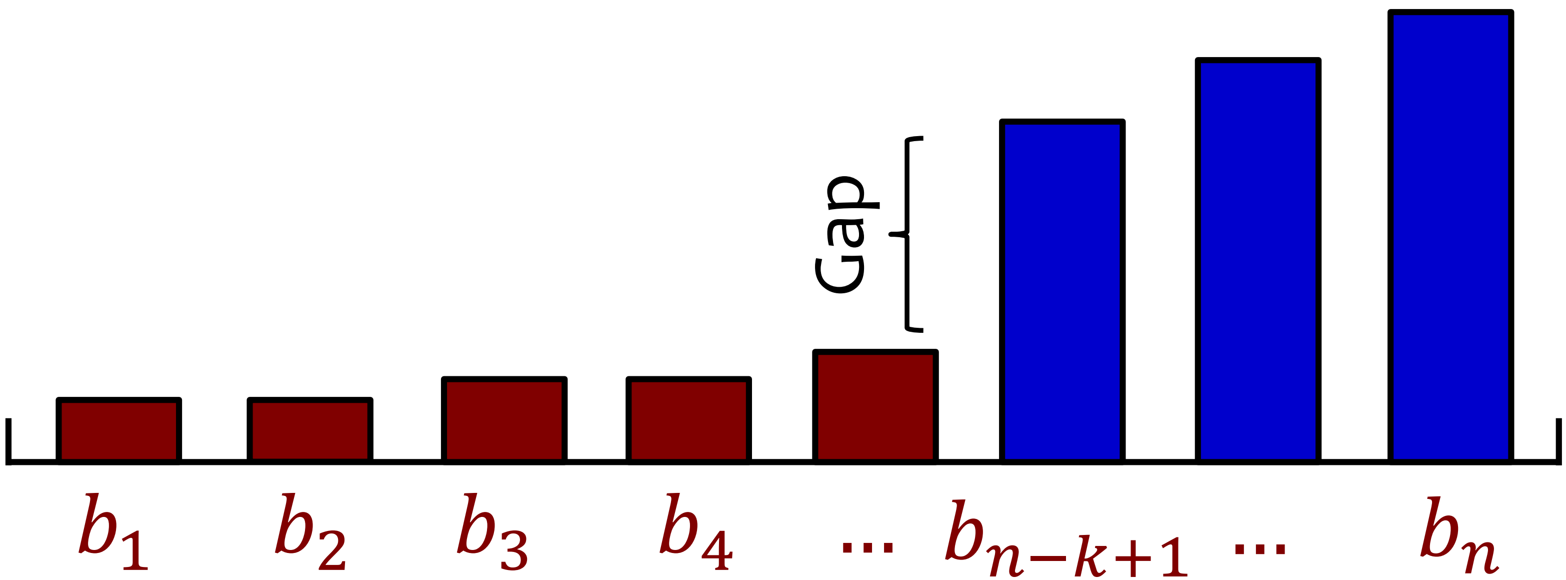
Exploiting gaps in basis lengths

Idea: Only match parity on “high order bits”.



Exploiting gaps in basis lengths

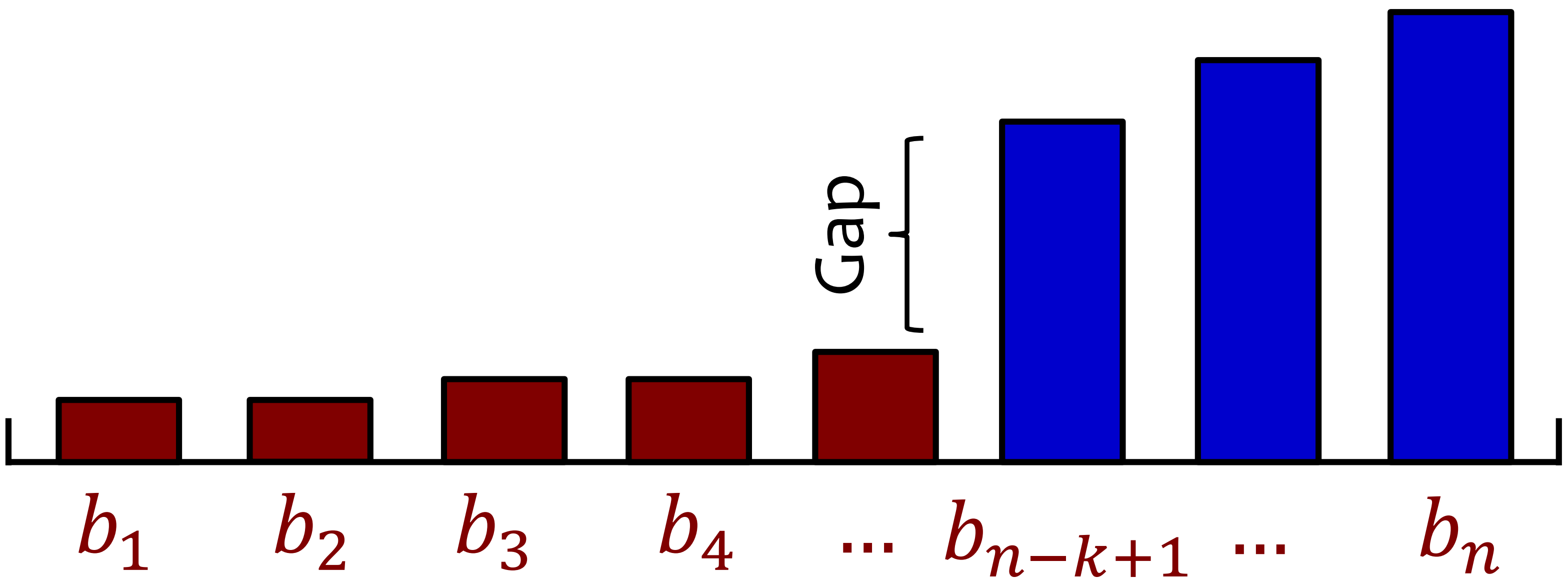
Idea: Only match parity on “high order bits”.



Parity of last k coefficients determines these coefficients exactly.

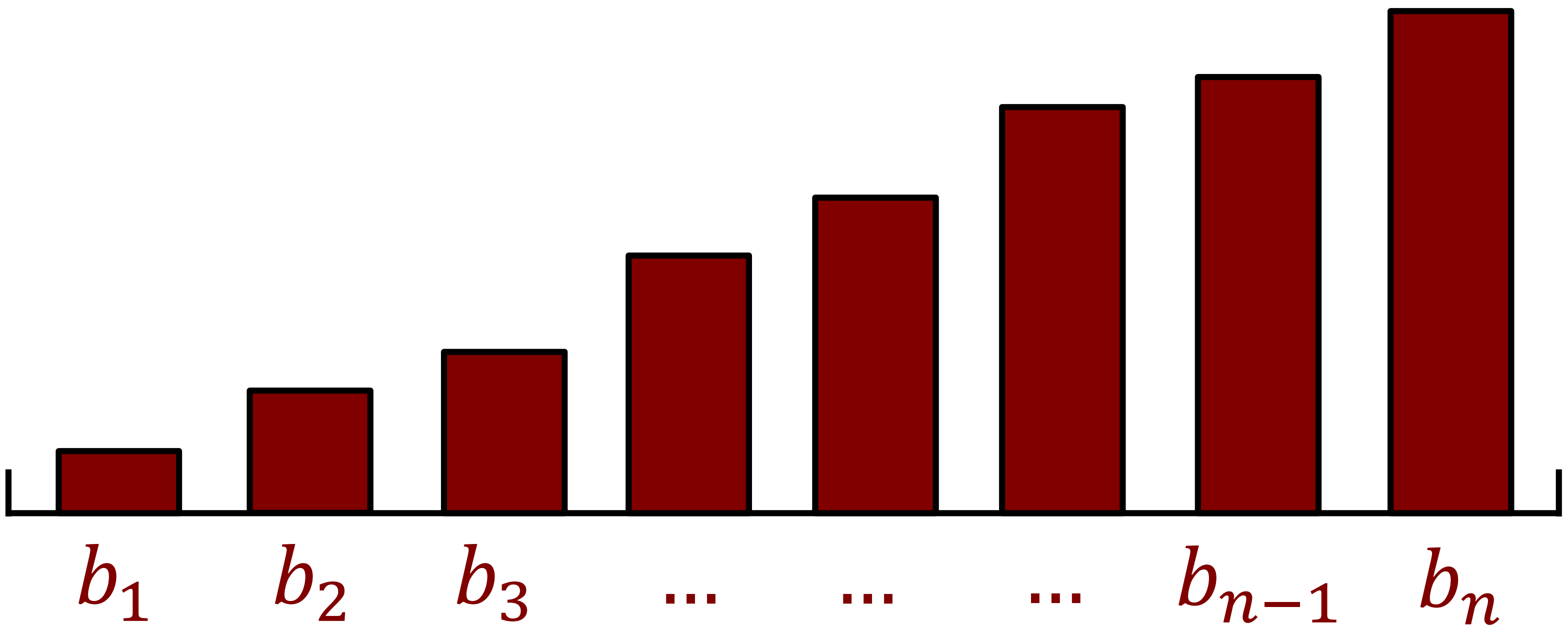
Exploiting gaps in basis lengths

Idea: Only match parity on "high order bits".



Idea: Can round first $n - k$ coefficients to desired parity without increasing distance to $\mathbf{t}$ by much.

Exploiting gaps in basis lengths

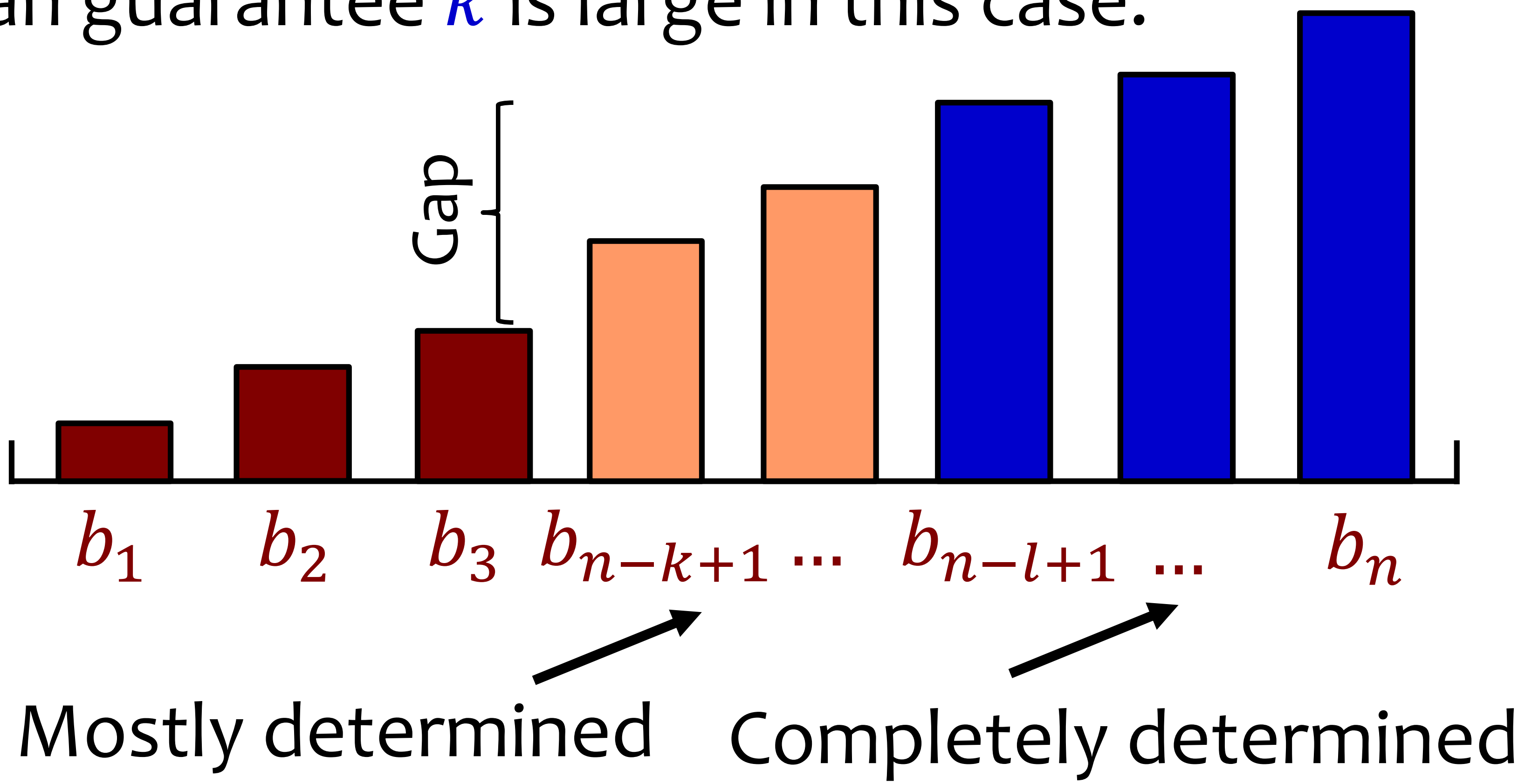


What if there are no large gaps?

Exploiting gaps in basis lengths

Idea: Again only match parity on last k bits.

Can guarantee k is large in this case.



High Level Algorithm

Input: n-dimensional lattice $\mathcal{L}$ and target $\mathbf{t}$.

Output: Closest lattice vectors in $\mathcal{L}$ to $\mathbf{t}$.

1. Compute **short** basis B of $\mathcal{L}$, and number k of "**high order coordinates**".
2. Get many $1 + 2^{-n}$ approx. closest vectors via **DGS**.
3. Group them according to last k coordinates with respect to B and recurse on each group .

Complexity Sketch

Initialization: (one shot $2^{n+o(n)}$ time)

Compute **short** basis B of $\mathcal{L}$, and number k of "**high order coordinates**" (can compute for each rec. level).

Per level work: ($2^{n+o(n)}$ time)

Sample many approx. closest vectors via **DGS**.

Recursion: ($\approx 2^k$ subproblems of dim. $n-k$)

Group them according to last k coordinates with respect to B and recurse.

Total runtime: $2^{n+o(n)}$

Key Challenges

Runtime:

1. Getting many **DGS** samples at low parameters.
2. Show last $\boldsymbol{k}$ **coeffs** $\approx$ determined by their parity.
3. Deal with $\approx \mathbf{2}^{\boldsymbol{k}}$ subproblems in recursion analysis.

Correctness:

Show that we **hit last** $\boldsymbol{k}$ **coeffs** of an exact closest vector with high probability.

Summary of Results

Discussed in this talk

- $2^{n+o(n)}$ algorithm for SVP and CVP.
- How to sample $2^{n/2}$ vectors from $D_{\mathcal{L},s}$ for any s in time $2^{n+o(n)}$

Additional results from this work

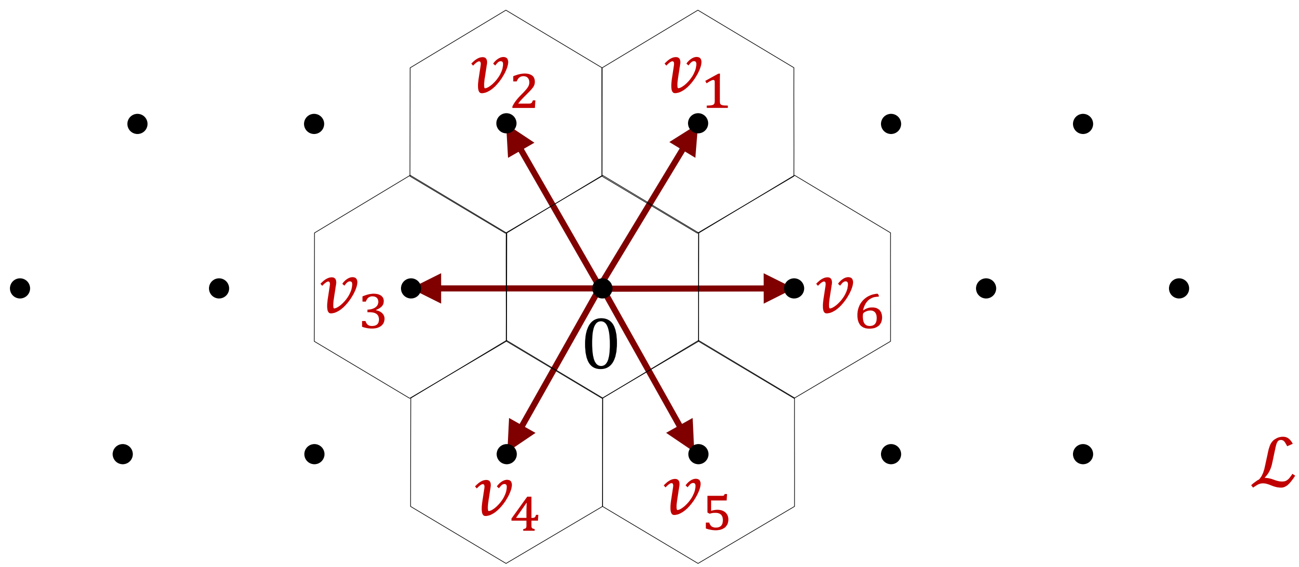
- $2^{n/2+o(n)}$ -time algorithm for sampling $2^{n/2}$ vectors above smoothing.
- 1.93-GapSVP.
- $.422$-BDD.

Recent work

- Sampling from DGS reduces to SVP. [Ste16]
  (not equivalence because the reduction in the other direction requires 1.38^n $D_{\mathcal{L},s}$ samples.)

Open Questions/Future Work

- Other uses for discrete Gaussian sampling at arbitrary parameters?
- Faster discrete centered Gaussian sampling?
- Strong lower bounds for CVP/SVP assuming SETH (or something similar)?
- Deterministic / Las Vegas algorithms with same complexity?



Thanks!