Alon Rosen

Discrete vs. Continuous Assumptions

One... two... five!
Three, sir.[*]

Learning parity with noise (LPN) shows that noise can make learning computationally hard. *Learning with errors* (LWE) extends noisy parity to arithmetic modulo $\mathbb{Z}_q$, with hardness tied to worst-case lattice problems. We distinguish recovering the secret from generating fresh samples with the same secret.

The chapter then passes to continuous distributions. *Continuous LWE* (CLWE) uses Gaussian inputs and a hidden unit direction; reduction modulo 1 retains the periodic structure behind the hardness. Homogeneous CLWE removes the labels, asking to distinguish a standard Gaussian from a planted Gaussian mixture.

Homogeneous CLWE (hCLWE) is naturally a hypothesis-testing problem. This leads to *statistical to computational gaps*: the sample size needed for detection in principle can be strictly smaller than the sample size needed for efficient detection. Cryptographic problems and Gaussian mixtures illustrate how far apart these thresholds can be.

The main points of this chapter are:

- Hard noisy learning problems, such as LPN, can serve as cryptographic assumptions.
- LWE extends noisy parity to arithmetic modulo q, and its average-case hardness is tied to worst-case lattice problems.
- Fresh LWE samples can be generated from old ones without recovering the secret; the leftover hash lemma explains why the new sample is nearly fresh.
- CLWE and hCLWE turn lattice-based hardness into continuous hidden-direction problems and structured Gaussian mixtures.
- Statistical and computational thresholds can be far apart; cryptographic and Gaussian-mixture examples illustrate the gap.

[*] Monty Python, *Monty Python and the Holy Grail*.

1 Learning parity with noise

We now present a learning problem that is conjectured to be cryptographically hard, together with some intuition for why. LPN is especially interesting because, for suitable noise rates, its hardness supports public-key encryption, which is not known to follow from one-way functions alone. Its arithmetic is entirely over $\mathbb{F}_2$, so the underlying operations are very fast bit operations.

Recall the class of linear functions from Chapter 1, now over $\mathbb{F}_2$. Fix a random unknown secret $\mathbf{s} \in \mathbb{F}_2^n$ and receive independent (sample, label) pairs

$$(\mathbf{a}, b), \qquad \mathbf{a} \leftarrow \mathbb{F}_2^n, \qquad b = \mathbf{a}^\top \mathbf{s} \pmod 2.$$

The task is to predict $\mathbf{a}^\top \mathbf{s}$ for a fresh random $\mathbf{a}$. Viewed as a learning problem, this is somewhat unusual by modern machine-learning conventions. The inputs and labels are bits, and the target is linear over $\mathbb{F}_2$, with addition and multiplication performed modulo 2. By contrast, standard linear models and neural-network training typically use real-valued arithmetic and parameters.

This alone is not cryptographic hardness. Stack m samples into a matrix A and vector $\mathbf{b}$. Then $\mathbf{b} = A\mathbf{s}$ over $\mathbb{F}_2$, and Gaussian elimination recovers $\mathbf{s}$ whenever the equations determine it uniquely. With $m = n + \lambda$ uniform samples, the probability that $\mathbf{s}$ fails to be determined is at most $2^{-\lambda}$.

LPN changes only the labels by adding independent random error bits.

Definition 1 (Learning parity with noise (LPN))**.** *Fix a noise rate $\eta \in (0, 1/2)$ and choose a secret $\mathbf{s} \leftarrow \mathbb{F}_2^n$. An LPN sample $(\mathbf{a}, b)$ is generated as*

$$\mathbf{a} \leftarrow \mathbb{F}_2^n, \qquad b = \mathbf{a}^\top \mathbf{s} + e \pmod 2,$$

where e is sampled from $\mathrm{Ber}(\eta)$; namely $e = 1$ with probability η and $e = 0$ with probability $1 - \eta$.

Search LPN asks to recover $\mathbf{s}$ from independent LPN samples. *Decisional LPN* asks to distinguish independent LPN samples from uniform pairs in $\mathbb{F}_2^n \times \mathbb{F}_2$.

With m independent LPN samples, there is enough information for the secret to be fully specified despite the noise. If $H_2(\eta) = -\eta \log_2 \eta - (1 - \eta) \log_2(1 - \eta)$ denotes binary entropy, the threshold is roughly

$$m > \frac{n}{1 - H_2(\eta)}.$$

For every constant $\eta < 1/2$, this is still only $\Theta(n)$ samples. LPN conjectures that recovery nevertheless remains computationally hard.

A first sanity check is whether guessing which equations are corrupted is indeed inefficient. With $m = \Theta(n)$ samples, the number of error patterns with the expected weight ηm is roughly

$$\binom{m}{\eta m} \approx 2^{mH_2(\eta)}.$$

If $\eta m = O(1)$, this is only polynomial. An even cheaper attack repeatedly chooses n equations and hopes that none are corrupted; one trial succeeds with probability $(1 - \eta)^n \approx e^{-\eta n}$. Hence $\eta = O(\log n / n)$ still permits polynomially many trials, while constant noise defeats this strategy.

Gaussian elimination does not directly survive the noise either. For a typical random full-rank $n \times n$ subsystem, each recovered secret bit depends on $\Theta(n)$ labels, since a typical row of A^{-1} is dense. If t error bits are combined, then

$$\Pr[e_1 + \cdots + e_t = 1] = \frac{1 - (1 - 2\eta)^t}{2},$$

so the bias from a fair coin is $(1-2\eta)^t$. For $t = \Theta(n)$ and constant noise, it is exponentially small.

There exist attacks that avoid this by canceling variables in blocks. For constant noise, they solve LPN in time $2^{O(n/\log n)}$ using $2^{O(n/\log n)}$ samples. Variants reduce the sample requirement to polynomial, at the price of running time $2^{O(n/\log\log n)}$. No polynomial-time algorithm is known for constant-noise LPN, a regime already sufficient for one-wayness.

At noise rate $\eta = n^{-1/2}$ with $2n$ samples, LPN hardness yields public-key encryption. This is a much rarer form of cryptographic structure than the unstructured hardness of a one-way function. For constant η, it remains open whether the standard polynomial-time LPN assumption alone implies public-key encryption. Public-key encryption is known from constant-noise LPN under the stronger assumption that LPN has $2^{\omega(\sqrt{n})}$ hardness.

2 Learning with errors

LPN gives noisy linear equations over a finite field. Learning with errors (LWE) uses the same template over $\mathbb{Z}_q$, with noise concentrated on small residues.

Definition 2 (Learning with errors (LWE))**.** *Fix an integer q and choose a secret $\mathbf{s} \leftarrow \mathbb{Z}_q^n$. An LWE sample $(\mathbf{a}, b)$ is generated as*

$$\mathbf{a} \leftarrow \mathbb{Z}_q^n, \qquad b = \mathbf{a}^\top \mathbf{s} + e \pmod q,$$

where e is drawn from a discrete Gaussian of width αq.

Search LWE asks to recover $\mathbf{s}$ from LWE samples. Decisional LWE asks to distinguish LWE samples from uniform pairs in $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

The main difference from LPN is the underlying arithmetic. LPN is over $\mathbb{F}_2$, and more generally can be instantiated over a finite field, whereas LWE is over the ring $\mathbb{Z}_q$. In $\mathbb{Z}_q$ we can talk about the magnitude of elements relative to q. This also makes sense in a prime field $\mathbb{F}_q$, but there is no canonical ordering or magnitude in an extension field $\mathbb{F}_{p^k}$. This change turns out to be important: LWE supports richer cryptography than is currently known from standard LPN, including fully homomorphic encryption, and it has a worst-case lattice foundation.

2.1 Why is LWE believed to be hard?

Reducing $\mathbf{a}^\top \mathbf{s}$ modulo q discards all bits but the k least significant ones, while the error e mainly perturbs the lowest bits. The sample is therefore a band of middle bits of $\mathbf{a}^\top \mathbf{s}$.

Without error, the samples satisfy $\mathbf{b} = \mathbf{A}\mathbf{s} \pmod q$, so the secret can be recovered using linear algebra. Without modular reduction, the samples satisfy $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$ over the integers or reals, so with enough samples the secret can be recovered using linear regression.

The strongest evidence for hardness comes from a connection to worst-case lattice problems. For a lattice $L(B) = \{B\mathbf{z} : \mathbf{z} \in \mathbb{Z}^n\}$, let $\lambda_1(L)$ be the length of its shortest nonzero vector. In γ-GapSVP, given B and d, the goal is to distinguish $\lambda_1(L) \le d$ from $\lambda_1(L) > \gamma d$ under this promise.

The best known algorithmic tradeoffs for γ-GapSVP roughly give $2^{O(k)}$ time for $\gamma = k^{n/k-1}$. For every constant γ, γ-GapSVP is NP-hard under randomized polynomial-time reductions. Under stronger time-complexity assumptions, hardness is known up to $n^{c/\log\log n}$ for some constant $c > 0$. Most relevant to LWE is a worst-case-to-average-case reduction.

Theorem 1 (Worst-case hardness of LWE)**.** *A polynomial-time algorithm for search LWE gives a quantum polynomial-time algorithm for worst-case $\widetilde{O}(n)$-GapSVP.*

More concretely, the approximation factor in the reduction is $\widetilde{O}(n/\alpha)$, where $\alpha q > 2\sqrt{n}$. Classical worst-case reductions are also known, with different parameters.

How hard are these worst-case problems? For the related γ-GapCVP, a recent Lean formalization claims NP-hardness up to $\gamma = n^{1/2-\varepsilon}$ for every fixed $\varepsilon > 0$. Pushing the inapproximability all the way to $C\sqrt{n}$ for large enough constant C would imply the astounding conclusion NP = coNP.

The LWE reduction works at the coarser $\widetilde{O}(n/\alpha)$ scale. This leaves a gap, but LWE is an average-case assumption: even a *heuristic* that works only on random LWE instances would refute it. The longer it resists such heuristic attacks, the more confidence we gain in its hardness.

2.2 Search, decision, and generation

In LWE, a polynomial-time *distinguisher* can be used to *recover* the secret $\mathbf{s}$ in polynomial time.

There is a different task that is weaker than recovery: after seeing many sample-label pairs, *generate* a fresh pair with the same hidden secret. This need not require learning $\mathbf{s}$.

Proposition 1. *Assuming LWE is hard, there are distributions whose hidden representation is hard to learn, yet fresh samples with the same representation can be efficiently generated.*

Proof. Given samples $(\mathbf{a}_i, b_i)$, let $\mathbf{A} = (\mathbf{a}_1, \dots, \mathbf{a}_m)$, $\mathbf{b} = (b_1, \dots, b_m)^\top$, and $\mathbf{e} = (e_1, \dots, e_m)^\top$. Choose $\mathbf{r} \leftarrow \{0,1\}^m$ and define

$$\mathbf{a}' = \mathbf{A}\mathbf{r} \pmod q, \qquad b' = \mathbf{b}^\top \mathbf{r} \pmod q, \qquad e' = \mathbf{e}^\top \mathbf{r} \pmod q.$$

Then

$$b' = \mathbf{a}'^\top \mathbf{s} + e' \pmod q.$$

Note that b' is a valid label for $\mathbf{a}'$ under the same secret $\mathbf{s}$. Since a typical $\mathbf{r}$ selects about $m/2$ samples, e' has width about $\sqrt{m/2}$ times the original width.

For $(\mathbf{a}', b')$ to be a truly fresh pair, $\mathbf{a}'$ must be close to uniform even conditioned on the original sample vectors $\mathbf{A} = (\mathbf{a}_1, \dots, \mathbf{a}_m)$.

Lemma 1 (Leftover hash lemma, specialized)**.** *If $m \geq n \log_2 q + 2\log_2(1/\varepsilon)$, then $(\mathbf{A}, \mathbf{a}')$ is ε-close in statistical distance to $(\mathbf{A}, \mathbf{u})$, where $\mathbf{u}$ is uniform in $\mathbb{Z}_q^n$ and independent of $\mathbf{A}$.*

Proof. The proof analyzes collision probability as a proxy for statistical closeness to uniform. Fix $\mathbf{A}$ and write $p_{\mathbf{A}}(\mathbf{x}) = \Pr_{\mathbf{r}}[\mathbf{A}\mathbf{r} = \mathbf{x}]$. For independent uniform $\mathbf{r}, \mathbf{r}'$, with $\mathbf{A}$ fixed,

$$\Pr_{\mathbf{r},\mathbf{r}'}[\mathbf{A}\mathbf{r} = \mathbf{A}\mathbf{r}'] = \sum_{\mathbf{x}} p_{\mathbf{A}}(\mathbf{x})^2.$$

To bound its expectation over $\mathbf{A}$, fix $\mathbf{r}, \mathbf{r}'$. If $\mathbf{r} = \mathbf{r}'$, which occurs with probability 2^{-m} over random $\mathbf{r}, \mathbf{r}'$, the collision is certain. Otherwise $\mathbf{d} = \mathbf{r} - \mathbf{r}'$ is nonzero, so choose a coordinate with $d_j = \pm 1$. Fixing all other columns of $\mathbf{A}$, exactly one value of the jth column makes $\mathbf{A}\mathbf{d} = 0$, so $\Pr_{\mathbf{A}}[\mathbf{A}\mathbf{r} = \mathbf{A}\mathbf{r}'] = q^{-n}$. Hence

$$\mathbb{E}_{\mathbf{A}}\left[\sum_{\mathbf{x}} p_{\mathbf{A}}(\mathbf{x})^2\right] \leq \frac{1}{q^n} + \frac{1}{2^m}.$$

Because $\mathbf{A}$ is revealed in both experiments, their joint statistical distance is the average distance of $p_{\mathbf{A}}$ from uniform. By Cauchy–Schwarz and the collision bound,

$$\Delta\big((\mathbf{A}, \mathbf{Ar}), (\mathbf{A}, \mathbf{u})\big) \le \frac{1}{2}\sqrt{q^n \, \mathbb{E}_{\mathbf{A}}\left[\sum_{\mathbf{x}} \left(p_{\mathbf{A}}(\mathbf{x}) - \frac{1}{q^n}\right)^2\right]} \le \frac{1}{2}\sqrt{\frac{q^n}{2^m}} \le \varepsilon.$$

□

Thus, up to statistical error ε, $(\mathbf{a}', b')$ has the distribution of a fresh LWE pair with secret $\mathbf{s}$ and the slightly wider noise distribution above. □

The construction of public-key encryption from LWE is closely related to Proposition 1. The public key consists of the original LWE samples $(\mathbf{a}_i, b_i)$, and the secret key is $\mathbf{s}$. To encrypt 0, generate a fresh LWE sample from the public samples; to encrypt 1, sample a uniform pair. Decryption uses $\mathbf{s}$ to distinguish the two cases.

3 Continuous learning with errors

The hard learning problems above are discrete. LPN over $\mathbb{F}_2$ and LWE over $\mathbb{Z}_q$. Learning is often over $\mathbb{R}^n$, with continuous inputs, parameters, and noise. We now build a continuous analogue of LWE while keeping its essential features: a hidden linear direction and modular reduction.

3.1 Continuous LWE

Continuous LWE keeps the structure of LWE while replacing the discrete quantities by continuous ones. The sample vector $\mathbf{a}$ is drawn from a standard Gaussian in $\mathbb{R}^n$, and the secret is a unit vector $\mathbf{w} \in \mathbb{R}^n$. The label is $\gamma \mathbf{a}^\top \mathbf{w}$ plus Gaussian noise, reduced modulo 1. Along the direction $\mathbf{w}$, the noiseless label is periodic with period $1/\gamma$, while β is the standard deviation of the noise.

Definition 3 (Continuous learning with errors (CLWE))**.** *Fix $\gamma, \beta > 0$ and choose a hidden unit vector $\mathbf{w} \leftarrow S^{n-1}$ uniformly. A CLWE sample $(\mathbf{a}, b)$ is generated as*

$$\mathbf{a} \leftarrow \mathcal{N}(0, I_n), \qquad b = \gamma \mathbf{a}^\top \mathbf{w} + e \pmod 1,$$

where e is drawn from a Gaussian of width β.

Search CLWE asks to recover $\mathbf{w}$ from CLWE samples. Decisional CLWE asks to distinguish samples from pairs $(\mathbf{a}, u)$ where $\mathbf{a} \leftarrow \mathcal{N}(0, I_n)$ and u is uniform in $[0, 1)$ and independent of $\mathbf{a}$.

Without modular reduction, $b = \gamma \mathbf{a}^\top \mathbf{w} + e$ is just linear regression, with γ rescaling $\mathbf{w}$. At fixed precision, the connection to modulus 1 is simple. A residue $z \bmod q$ keeps the k least significant bits of z, and dividing by q moves those bits after the binary point. Viewing a fixed-precision real number in binary makes it apparent that CLWE is just LWE in disguise. Indeed:

Theorem 2 (Hardness of CLWE)**.** *A polynomial-time algorithm for decisional CLWE gives a polynomial-time algorithm for decisional LWE.*

CLWE is conjectured to be hard in a regime where γ is at least on the order of $\sqrt{n}$, the noise satisfies $\beta \in (0, 1)$, and $\gamma/\beta \le \mathrm{poly}(n)$. The noise cannot be taken arbitrarily small: nearly noiseless CLWE, $\beta \approx 0$, is solvable in polynomial time using lattice reduction.

3.2 Homogeneous CLWE

Homogeneous CLWE (hCLWE) is a hypothesis-testing problem between a standard Gaussian and a distribution that is Gaussian in every direction but one. Parameters γ, β are as in CLWE.

Definition 4 (Homogeneous CLWE)**.** *Given m samples $\mathbf{a}_1, \ldots, \mathbf{a}_m \in \mathbb{R}^n$, distinguish between:*

- Null distribution: $\mathbf{a}_i \leftarrow \mathcal{N}(0, I_n)$.
- Planted distribution: *choose a hidden unit vector $\mathbf{w} \leftarrow S^{n-1}$ uniformly. In every direction orthogonal to $\mathbf{w}$, the samples are standard Gaussian, while the scalar projection $a_i^{\parallel} = \mathbf{a}_i^\top \mathbf{w}$ is a noisy discrete Gaussian concentrated near points k/γ, for $k \in \mathbb{Z}$, with peak width β/γ.*

Hardness is inherited from CLWE via rejection sampling: keep CLWE samples whose labels fall near 0, and then discard the labels. Under the null this leaves the Gaussian input distribution unchanged, while in the planted case it creates the periodic structure above.

Theorem 3 (Hardness of hCLWE)**.** *A polynomial-time algorithm for hCLWE gives a polynomial-time algorithm for decisional CLWE, and hence for decisional LWE.*

The difficulty is finding the hidden direction. If $\mathbf{w}$ were known, the problem would be one-dimensional: project onto $\mathbf{w}$ and test whether $\gamma \mathbf{a}^\top \mathbf{w} \pmod 1 \approx 0$.

The planted distribution is a mixture of Gaussians. A Gaussian mixture is obtained by first choosing one of several Gaussian distributions and then drawing a sample from it.

Proposition 2. *Assuming hCLWE is hard, there are Gaussian mixtures that are hard to distinguish from a standard Gaussian in polynomial time.*

Proof. Write $\mathbf{a} = a^{\parallel}\mathbf{w} + \mathbf{a}^{\perp}$, where $a^{\parallel} = \mathbf{a}^\top \mathbf{w}$ is the one-dimensional component along $\mathbf{w}$, and $\mathbf{a}^{\perp} = \mathbf{a} - (\mathbf{a}^\top \mathbf{w})\mathbf{w}$ is the $(n-1)$-dimensional component orthogonal to $\mathbf{w}$. Under the planted distribution, conditioned on the integer k, $a^{\parallel}$ is Gaussian near k/γ and $\mathbf{a}^{\perp}$ is an independent standard Gaussian. Thus $\mathbf{a}$ is Gaussian conditioned on k, with mean near $(k/\gamma)\mathbf{w}$; averaging over k gives a Gaussian mixture. Under the null, $\mathbf{a}$ is a single standard Gaussian. Hence distinguishing this mixture from a standard Gaussian would solve hCLWE. □

3.3 Statistical to computational gaps

The hCLWE problem is a specific instance of a hypothesis testing question:

- H_0: Null. The m samples contain no shared signal.
- H_1: Planted. Choose a hidden signal, then draw all m samples using that same signal.

Detection distinguishes H_0 from H_1; *recovery* reconstructs the signal. For fixed signal-to-noise ratio and as m grows, the statistical threshold m_{stat} is the sample scale at which detection becomes information-theoretically possible; below it, there is no discernible information about the signal.

For LPN and LWE, H_0 is uniform sample-label pairs and H_1 uses one hidden secret $\mathbf{s}$ to generate the noisy labels. For CLWE, H_0 has Gaussian inputs with independent uniform labels and H_1 uses one hidden direction $\mathbf{w}$ to generate labels with Gaussian noise; hCLWE plants the same direction directly in the sample distribution. The statistical threshold m_{stat} is roughly linear in n: $\Theta(n)$ for LPN, and up to logarithmic factors also for LWE, CLWE, and hCLWE.

Some problems have a threshold m_{comp}, above which detection or recovery is computationally easy. A *statistical to computational gap* is the regime $m_{\text{stat}} \ll m_{\text{comp}}$: the data contain the signal, but efficient algorithms still cannot extract it.

Proposition 3. *Assuming LWE (similarly LPN) is hard, there exist hypothesis testing problems with no polynomial threshold m_{comp}.*

Proof. Proposition 1 enables us to generate fresh LWE samples (similarly for LPN) on our own, without knowing the secret; if a polynomial m_{comp} existed, we could generate enough samples to cross it, rendering the problem (with some loss in noise rate) easy. □

For some non-periodic hidden-direction Gaussian mixtures, the kth empirical moment gives an $n^{O(k)}$-time distinguisher: matching the first $k-1$ Gaussian moments along $\mathbf{w}$ hides all lower-degree statistics, while the kth reveals $\mathbf{w}$. Its n^k-entry empirical tensor takes $n^{O(k)}$ time to compute. On the other hand, no faster general method is known, and statistical-query lower bounds suggest optimality. This leads to the conjecture that $m_{\text{comp}} = n^{\Theta(k)}$ while $m_{\text{stat}} = O(n)$.

Other standard examples of such gaps include planted clique, sparse PCA, tensor PCA, and planted dense subgraph. Any secure cryptographic pseudorandom generator with seed size n and output length ℓ has $m_{\text{comp}} > \ell$, where the samples are its individual output bits.

4 Historical notes

Cryptography from hard learning goes back to [BFKL93]. For LPN, [BKW03, Lyu05] give the subexponential attacks above and [Ale03, YZ16] the public-key constructions.

Regev introduced LWE and gave a quantum worst-case reduction from GapSVP [Reg09]; classical reductions followed [BLPRS13]. Under stronger time-complexity assumptions, Haviv and Regev proved SVP hardness up to factor $n^{c/\log\log n}$ for some constant $c > 0$ [HR12]. The leftover hash lemma, central to modern cryptography, was introduced by Impagliazzo, Levin and Luby in their construction of pseudorandom generators from one-way functions [ILL89].

Bruna, Regev, Song and Tang introduced CLWE and Gaussian pancakes and connected them to hardness of learning Gaussian mixtures [BRST21]. Diakonikolas, Kane and Stewart established statistical-query gaps for high-dimensional Gaussian-mixture inference [DKS17]. Gupte, Vafa and Vaikuntanathan later showed that CLWE is as hard as LWE via direct reductions and strengthened the resulting hardness for Gaussian-mixture density estimation [GVV22].

In summer 2026, a frontier language model proved $n^{1/400}$-factor hardness for GapCVP [Ope26]; a subsequent Lean formalization (`https://github.com/Mira-acc/cvp`) claims a strengthening of this to $n^{1/2-\varepsilon}$ for every fixed $\varepsilon > 0$. The square-root-scale upper bounds are due to Goldreich and Goldwasser [GG98], and Aharonov and Regev [AR05].

References

[Ale03] M. Alekhnovich. More on average case vs approximation complexity. In *FOCS*, 2003.

[AR05] D. Aharonov and O. Regev. Lattice problems in NP ∩ coNP. *Journal of the ACM*, 52(5):749–765, 2005.

[BFKL93] A. Blum, M. L. Furst, M. J. Kearns, and R. J. Lipton. Cryptographic primitives based on hard learning problems. In *CRYPTO*, 1993.

[BKW03] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *JACM*, 50(4):506–519, 2003.

[BLPRS13] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In *STOC*, pages 575–584, 2013.

[BRST21] J. Bruna, O. Regev, M. J. Song, and Y. Tang. Continuous LWE. In *STOC*, pages 694–707, 2021.

[DKS17] I. Diakonikolas, D. M. Kane, and A. Stewart. Statistical query lower bounds for robust estimation of high-dimensional Gaussians and Gaussian mixtures. In *FOCS*, 2017.

[GVV22] A. Gupte, N. Vafa, and V. Vaikuntanathan. Continuous LWE is as hard as LWE and applications to learning Gaussian mixtures. arXiv:2204.02550, 2022.

[GG98] O. Goldreich and S. Goldwasser. On the limits of non-approximability of lattice problems. In *STOC*, pages 1–9, 1998.

[HR12] I. Haviv and O. Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. *Theory of Computing*, 8(23):513–531, 2012.

[ILL89] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random number generation from one-way functions. In *STOC*, pages 12–24, 1989.

[Lyu05] V. Lyubashevsky. The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem. In *RANDOM*, 2005.

[Ope26] OpenAI. Ten advances in mathematics and theoretical computer science. Manuscript, updated August 6, 2026.

[Reg09] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6), 2009.

[YZ16] Y. Yu and J. Zhang. Cryptography with auxiliary input and trapdoor from constant-noise lpn. In *CRYPTO*, 2016.