Fine-grained hardness of CVP(P)—Everything that we can prove (and nothing else)

Divesh Aggarwal*
dcsdiva@nus.edu.sg

Huck Bennett†
huckbennett@gmail.com

Alexander Golovnev‡
alexgolovnev@gmail.com

Noah Stephens-Davidowitz§
noahsd@gmail.com

Abstract

We show that the Closest Vector Problem in the ℓ_p norm (CVP_p) cannot be solved in $2^{(1-\varepsilon)n}$ time for all $p \notin 2\mathbb{Z}$ and $\varepsilon > 0$ (assuming SETH). In fact, we show that the same holds even for (1) the approximate version of the problem (assuming a gap version of SETH); and (2) CVP_p with preprocessing, in which we are allowed arbitrary advice about the lattice (assuming a non-uniform version of SETH). For "plain" CVP_p, the same hardness result was shown in [Bennett, Golovnev, and Stephens-Davidowitz FOCS 2017] for all but finitely many $p \notin 2\mathbb{Z}$, where the set of exceptions depended on ε and was not explicit. For the approximate and preprocessing problems, only very weak bounds were known prior to this work.

We also show that the restriction to $p \notin 2\mathbb{Z}$ is in some sense inherent. In particular, we show that no "natural" reduction can rule out even a $2^{3n/4}$-time algorithm for CVP_2 under SETH. For this, we prove that the possible sets of closest lattice vectors to a target in the ℓ_2 norm have quite rigid structure, which essentially prevents them from being as expressive as 3-CNFs.

*National University of Singapore.
†University of Michigan.
‡Harvard University.
§Massachusetts Institute of Technology.

1 Introduction

A lattice $\mathcal{L}$ is the set of all integer linear combinations of linearly independent basis vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{R}^d$,

$$\mathcal{L} = \mathcal{L}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) := \{z_1\boldsymbol{b}_1 + \cdots + z_n\boldsymbol{b}_n \ : \ z_i \in \mathbb{Z}\} \ .$$

We call n the *rank* of the lattice $\mathcal{L}$ and d the *dimension* or the *ambient dimension* of the lattice.

The two most important computational problems on lattices are the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP). Given a basis for a lattice $\mathcal{L} \subset \mathbb{R}^d$, SVP asks us to compute the minimal length of a non-zero vector in $\mathcal{L}$, and CVP asks us to compute the distance from some target point $\boldsymbol{t} \in \mathbb{R}^d$ to the lattice. Typically, we define length and distance in terms of the ℓ_p norm for some $1 \leq p \leq \infty$, given by

$$\|\boldsymbol{x}\|_p := (|x_1|^p + |x_2|^p + \cdots + |x_d|^p)^{1/p}$$

for finite p and

$$\|\boldsymbol{x}\|_\infty := \max_{1 \leq i \leq d} |x_i| \ .$$

In particular, the case where $p = 2$ corresponds to the Euclidean norm, which is the most important and best-studied in this context. We write SVP_p and CVP_p for the respective problems in the ℓ_p norm. CVP is known to be at least as hard as SVP (in any norm, under an efficient reduction that preserves the rank and approximation factor) [GMSS99] and appears to be significantly harder.

In the past decade, these problems have taken on still more importance, as their hardness underlies the security of most post-quantum public-key cryptography schemes, while the schemes that are currently used for most practical applications are not secure against quantum computers. Recent rapid progress in quantum computing (e.g., [A+19]) has therefore created a rush to switch to lattice-based cryptography in many applications. Indeed, for this reason, lattice-based cryptography is in the process of standardization for widespread use [NIS16].

Given the obvious importance of these problems, they have been studied quite extensively. However, in spite of much effort algorithmic progress has stalled for CVP. The fastest algorithm for CVP_2 runs in $2^{n+o(n)}$ time [ADS15]—even for arbitrarily large constant approximation factors—and there are fundamental reasons that our current techniques cannot do better.[1] For arbitrary p, the fastest known exact algorithm is still Kannan's $n^{O(n)}$-time algorithm from over thirty years ago [Kan87]. For constant-factor approximation and arbitrary p, Blömer and Naewe [BN09] gave a $2^{O(d)}$-time algorithm, which was later improved to $2^{O(n)}$ time by Dadush [Dad12], and a $4^{(1+\varepsilon)d}$-time algorithm for $p = \infty$ by Aggarwal and Mukhopadhyay [AM18].

While we have known for decades that CVP_p is NP-hard [vEB81], even to approximate [DKRS03], such coarse hardness results are insufficient to rule out, e.g., a $2^{n/20}$-time algorithm or even a $2^{\sqrt{n}}$-time algorithm. If such algorithms were found, they would have innumerable positive applications, but they would also render current lattice-based cryptographic constructions broken in practice. Even a small improvement beyond 2^n time would have major consequences.

[1] There are only two algorithms that solve CVP_2 in its exact form in time $2^{O(n)}$ [MV13, ADS15], and both of them involve enumeration over all 2^n cosets of $\mathcal{L}$ modulo $2\mathcal{L}$. (These cosets arise naturally in this context, and they play a large role in Section 6.) There are other approaches that achieve constant-factor approximation in time $2^{O(n)}$, but the constant in the exponent is significantly larger. The situation for SVP is far more dynamic. See, e.g., [BDGL16, AS18b].

In [BGS17], we therefore initiated the study of the *fine-grained hardness* of CVP in an effort to explain this lack of algorithmic progress and to give evidence for the *quantitative* security of lattice-based cryptography. We showed that there is no $2^{(1-\varepsilon)n}$-time algorithm for CVP_p assuming the Strong Exponential Time Hypothesis (SETH, a widely believed conjecture in complexity theory, defined in Section 2), but we were only able to prove this lower bound explicitly for odd integers p (and $p = \infty$). For other values of p, our result was much weaker. For every $\varepsilon > 0$, we showed that there are at most finitely many $p \notin 2\mathbb{Z}$ with a $2^{(1-\varepsilon)n}$-time algorithm for CVP_p (assuming SETH). In particular, for any *specific* value of $p \notin (2\mathbb{Z}+1) \cup \{\infty\}$, we could not rule out such an algorithm. (We did, however, rule out $2^{o(n)}$-time algorithms for all p.)

Furthermore, our results were far weaker for two important variants of the problem. First, for the near-exact version of the problem (i.e., the problem of approximating CVP_p up to some constant factor), we were only able to rule out $2^{o(n)}$-time algorithms. Second, our lower bounds were quite weak for the problem of CVP_p with preprocessing (CVPP_p), an offline-online variant of CVP_p where an unbounded-time preprocessing algorithm may perform arbitrary preprocessing on the lattice $\mathcal{L}$ in a way that helps an online query algorithm to find a closest lattice vector to a given target $\boldsymbol{t} \in \mathbb{R}^d$. In [BGS17], we were only able to rule out a $2^{o(\sqrt{n})}$-time algorithm for this problem. It therefore remained plausible that much faster algorithms could exist for CVPP_p than for CVP_p or for constant-factor approximate CVP_p. Such algorithms would, for example, lead to very strong attacks on certain lattice-based cryptographic schemes.

In follow-up work, we used the main result of [BGS17] to prove strong lower bounds for SVP [AS18a] and also for SIVP [AC19]. However, these works inherited some of the deficiencies described above. Specifically, the strongest hardness results in both works only applied to odd integers $p \in (2\mathbb{Z}+1)$ (and $p = \infty$) and some non-explicit set of additional p.

1.1 Our results

Our first main result is an extension of the main result in [BGS17] to *all* p except for the even integers, *and* to CVPP_p and approximate CVP_p. (See Table 1. In the introduction, we informally refer to an "approximate variant of SETH" as Gap-SETH. See Definition 2.7 for a formal definition due to Manurangsi [Man19].)

Theorem 1.1 (Informal)**.** *For every $1 \le p \le \infty$ with $p \notin 2\mathbb{Z}$, there is no $2^{(1-\varepsilon)n}$-time algorithm for CVP_p for any constant $\varepsilon > 0$ unless SETH is false. The same conclusion holds for CVPP_p unless non-uniform SETH is false.*

Furthermore, for every $1 \le p \le \infty$ with $p \notin 2\mathbb{Z}$ and constant $\varepsilon > 0$, there is no $2^{(1-\varepsilon)n}$-time algorithm for γ_ε-approximate CVP_p for some $\gamma_\varepsilon > 1$ unless Gap-SETH is false.

As in [BGS17], our result is actually a bit stronger than the above. SETH-based hardness only requires a reduction from k-SAT to CVP_p, but we show a reduction from Max-k-SAT, and even from weighted Max-k-SAT.

In fact, we also rule out $2^{o(n)}$-time algorithms for CVPP_p under a weaker complexity-theoretic assumption: the (non-uniform) Exponential Time Hypothesis. This weaker lower bound under a weaker assumption holds for all $p \neq 2$—including even integers $p \ge 4$.

Theorem 1.1 also yields immediate similar improvements to the hardness of SVP_p and SIVP_p, i.e., to the results of [AS18a, AC19]. In particular, by the main results in [AC19], the 2^n hardness for CVP_p and its approximate variant immediately extends to SIVP_p. The results for SVP_p are rather complicated, as they vary with p in complex ways [AS18a], but our results imply extensions

of [AS18a] to more values of p than were known previously. See Appendix A for a complete statement of the result.

The restriction that p is not an even integer is unfortunate, especially because we are most interested in the case when $p = 2$. But, this seems inherent. (In fact, it is known that ℓ_2 is "the easiest norm" in a certain precise sense [RR06].) Indeed, in [BGS17], we already showed that our specific techniques are insufficient to prove hardness for $p \in 2\mathbb{Z}$.

Here, we also rule out a far more general class of techniques for $p = 2$, which we call "natural reductions." These are reductions with a bijection between witnesses. Specifically, a reduction from a k-SAT formula ϕ to CVP_p over a lattice with basis $\mathbf{B}$ is natural if there is a fixed (not necessarily efficient) mapping $f : \{0,1\}^n \to \mathbb{Z}^{n'}$ such that $\mathbf{B}f(\boldsymbol{z})$ is a closest lattice vector if and only if $\boldsymbol{x}$ is a satisfying assignment (assuming that ϕ is satisfiable). We also mention here the fact that natural reductions cannot prove better than 2^n hardness for $1 < p < \infty$. We include a simple proof of this fact in Section 1.3.

Theorem 1.2 (Informal)**.** *There is no natural reduction from 3-SAT on n variables to* CVP_2 *on a lattice with rank $n' \leq 4(n-2)/3$. In particular, no natural reduction can rule out even a $2^{3n/4}$-time algorithm for* CVP_2 *under SETH.*

Furthermore, for any $1 < p < \infty$, there is no natural reduction from 3-SAT on n variables to CVP_p *on a lattice with rank $n' < n$. In particular, no natural reduction can rule out a 2^n-time algorithm for* CVP_p *under SETH for $1 < p < \infty$.*

Notice that we even rule out reductions from 3-SAT to CVP. To prove SETH-hardness, we would need to show a reduction from k-SAT for all constant $k \geq 3$.

Behind (the non-trivial $p = 2$ part of) Theorem 1.2 are two new techniques. First is a new result concerning the structure of the closest lattice vectors to a target point in the ℓ_2 norm. Specifically, we show that the structure of the closest vectors is quite rigid modulo $2\mathcal{L}$. Second is a new and tighter proof of Szemerédi's cube lemma for the boolean hypercube. We expect both of these results to be of independent interest.

1.2 Our reductions

The high-level idea behind our reductions (and those of [BGS17]) is as follows. The reduction is given as input a list $\phi_1, \ldots, \phi_m$ of k-clauses on n boolean variables $x_1, \ldots, x_n$, where $k \geq 2$ is some constant. We wish to construct some basis $\mathbf{B} \in \mathbb{R}^{d \times n}$ and target $\boldsymbol{t} \in \mathbb{R}^d$ such that for any $\boldsymbol{z} \in \mathbb{Z}^n$, $\|\mathbf{B}\boldsymbol{z} - \boldsymbol{t}\|_p^p$ for $\boldsymbol{z} \in \mathbb{Z}^n$ is small if and only if $\boldsymbol{z} \in \{0,1\}^n$ represents an assignment that satisfies all of the ϕ_i.

To that end, for each ϕ_i, we wish to find a matrix $\Phi_i \in \mathbb{R}^{d' \times n}$ and target $\boldsymbol{t}_i \in \mathbb{R}^{d'}$ such that $\|\Phi_i \boldsymbol{z} - \boldsymbol{t}_i\|_p^p$ is small if and only if $z_{j_1}, \ldots, z_{j_k} \in \{0,1\}$ represents an assignment that satisfies ϕ_i. If we could find such matrices, we could take

$$\mathbf{B} := \begin{pmatrix} \Phi_1 \\ \Phi_2 \\ \vdots \\ \Phi_m \\ 2\alpha I_n \end{pmatrix} \in \mathbb{R}^{md' \times n} \qquad \boldsymbol{t} := \begin{pmatrix} \boldsymbol{t}_1 \\ \boldsymbol{t}_2 \\ \vdots \\ \boldsymbol{t}_m \\ \alpha \mathbf{1} \end{pmatrix}, \tag{1}$$

Problem		Upper bounds		Lower bounds	
		Exact	Approximate	Exact	Approximate
CVP_p	$p \notin 2\mathbb{Z}$	$n^{O(n)}$	$2^{O(n)}$	$2^{(1-\varepsilon)n}*$	$2^{(1-\varepsilon)n}$
	$p \neq 2$	$n^{O(n)}$	$2^{O(n)}$	$2^{\Omega(n)}$	$2^{\Omega(n)}$
	$p = 2$	$2^{n+o(n)}$	$2^{n+o(n)}$	$2^{\Omega(n)}$	$2^{\Omega(n)}$
CVPP_p	$p \notin 2\mathbb{Z}$	$n^{O(n)}$	$2^{O(n)}$	$2^{(1-\varepsilon)n}$	—
	$p \neq 2$	$n^{O(n)}$	$2^{O(n)}$	$2^{\Omega(n)}$	—
	$p = 2$	$2^{n+o(n)}$	$2^{n+o(n)}$	$2^{\Omega(\sqrt{n})}$	—

Table 1: A summary of known quantitative upper and lower bounds under various assumptions on the complexity of CVP_p and CVPP_p for $p \in [1, \infty]$. New results appear in blue (with a star next to the one result that is only novel for some p). Upper bounds for the approximate problems are for any constant approximation factor $\gamma > 1$, while lower bounds are for some small, explicit approximation factor $\gamma > 1$ depending on p (and, in the case of CVP_p for $p \notin 2\mathbb{Z}$, also on $\varepsilon > 0$). The $2^{(1-\varepsilon)n}$-time lower bounds are based on SETH (or Gap-SETH or non-uniform SETH), while the $2^{\Omega(\sqrt{n})}$-time and $2^{\Omega(n)}$-time lower bounds are based on ETH (or Gap-ETH or non-uniform ETH).

where $\alpha\mathbf{1} \in \mathbb{R}^n$ is the vector whose coordinates are all α. Then, $\|\mathbf{B}\boldsymbol{z} - \boldsymbol{t}\|_p^p = \sum_i \|\Phi_i \boldsymbol{z} - \boldsymbol{t}_i\|_p^p$ will be small if and only if $\boldsymbol{z} \in \{0,1\}^n$ corresponds to a satisfying assignment. (By taking α to be sufficiently large, we can guarantee that any closest vectors must be of the form $\mathbf{B}\boldsymbol{z}$ for $\boldsymbol{z} \in \{0,1\}^n$.)

Since $\Phi_i\{0,1\}^n - \boldsymbol{t}_i = \{\Phi_i \boldsymbol{z} - \boldsymbol{t}_i \ : \ \boldsymbol{z} \in \{0,1\}^n\}$ is a parallelepiped, and since the most important case (corresponding to k-SAT) is when all but one point in this set is long and all others are short, we call such objects *isolating parallelepipeds*, as we explain below. The difficult step in these reductions is therefore to find isolating parallelepipeds $\Phi_i, \boldsymbol{t}_i$.

Finding isolating parallelepipeds. We say that a parallelepiped $\Phi\{0,1\}^k - \boldsymbol{t}$ is a (p,k)-*isolating parallelepiped* if all $\|\Phi\boldsymbol{z} - \boldsymbol{t}\|_p = 1$ for non-zero $\boldsymbol{z} \in \{0,1\}^k$ and $\|\Phi\mathbf{0} - \boldsymbol{t}\|_p = \|\boldsymbol{t}\|_p > 1$. (We think of the vertex $-\boldsymbol{t}$ as "isolated" from the others. See Figure 1.) To find isolating parallelepipeds, we construct a family of parallelepipeds parameterized by $\alpha_1, \ldots, \alpha_{2^k} \geq 0$ and $t^* \in \mathbb{R}$. This family has the useful property that the norms $\|\Phi\boldsymbol{z} - \boldsymbol{t}\|_p^p$ are linear in the α_i for fixed t^*. (In [BGS17], we used a less general family of parallelepipeds.)

So, finding isolating parallelepipeds essentially reduces to showing that a certain system of linear equations has a solution. (We actually need a non-negative solution, but we ignore this technical issue in the introduction.) To that end, we study the matrix $H_{k,p}(t^*) \in \mathbb{R}^{2^k \times 2^k}$ corresponding to this system of linear equations and try to show that its determinant is non-zero for some computable choice of t^*. To do this, we observe that $H_{k,p}(t^*)$ satisfies the recurrence

$$H_{k,p}(t^*) = \begin{pmatrix} H_{k-1,p}(t^* - 1) & H_{k-1,p}(t^* + 1) \\ H_{k-1,p}(t^* + 1) & H_{k-1,p}(t^* - 1) \end{pmatrix}.$$

(It is this recurrence that makes this family more useful than the less general family in [BGS17].) This makes showing that $\det(H_{k,p}(t^*))$ is non-zero susceptible to a proof by induction on k.

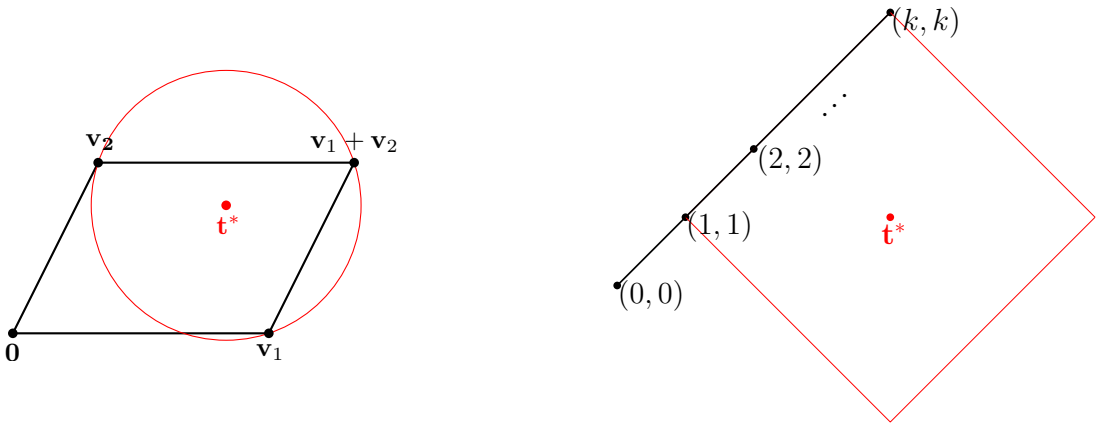

Figure 1: (p,k)-isolating parallelepipeds for $p = 2, k = 2$ (left) and $p = 1, k \geq 1$ (right). On the left, the vectors $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, and $\boldsymbol{v}_1 + \boldsymbol{v}_2$ are all at the same distance from $\boldsymbol{t}^*$, while $\mathbf{0}$ is strictly farther away. On the right is the degenerate parallelepiped generated by k copies of the vector $(1,1)$. The vectors (i,i) are all at the same ℓ_1 distance from $\boldsymbol{t}^*$ for $1 \leq i \leq m$, while $(0,0)$ is strictly farther away. The (scaled) unit balls centered at $\boldsymbol{t}^*$ are shown in red, while the parallelepipeds are shown in black. (Figure taken from [BGS17].)

To that end, we use a formula for the determinant of block matrices of this form to show by induction that $\det(H_{k,p}(t^*))$ is equal to the product of 2^k functions of t^*. These functions are in turn each non-zero $\mathbb{R}$-linear combinations of functions of the form $(t^* + \beta)^p$ for distinct $\beta \in \mathbb{R}$. (The determinant is actually a piecewise combination of such functions, but we ignore this here.) We prove that such functions are $\mathbb{R}$-linearly independent if (and only if) either $p \geq k$ or $p \notin \mathbb{Z}$. Therefore, the functions cannot be identically zero for such p, which in turn implies that $\det(H_{k,p}(t^*))$ is not identically zero as a function of t^*, as needed. We finish the proof by noting that $\det(H_{k,p}(t^*))$ is (piecewise) analytic so that its zeros must be isolated, and it therefore has a computable non-zero point.

By combining this construction with our previous work, we completely characterize the values of p and k for which (p,k)-isolating parallelepipeds exist. Namely, the only case not handled by the construction above is the case where $p \in \{1, \ldots, k-1\}$. In this case, [BGS17] showed that such parallelepipeds exist for odd p but cannot exist for even $p < k$. (We provide a full proof of this latter claim in Lemma 6.1.) So, (p,k)-isolating parallelepipeds exist if and only if $p \notin \{2i \ : \ i < k/2\}$.

As a corollary, we show a reduction from (weighted Max-)k-SAT on n variables to a CVP_p instance with rank n for all $p \notin \{2i \ : \ i < k/2\}$. In particular, we prove that CVP_p is SETH-hard for all $p \notin 2\mathbb{Z}$.

Hardness of CVPP_p. We next show how to extend the hardness result above from CVP_p to the Closest Vector Problems with Preprocessing in the ℓ_p norm (CVPP_p). Namely, we show that CVPP_p is 2^n-hard assuming (non-uniform) SETH for all $p \notin 2\mathbb{Z}$. To do this, we define an enhanced notion of an isolating parallelepiped, that we call an *on-off-isolating parallelepiped* (this is analogous to what [SV19] does for codes). An on-off-isolating parallelepiped is an isolating parallelepiped $\Phi, \boldsymbol{t}^*$ together with a target $\boldsymbol{t}_{\text{off}}$ such that $\|\Phi \boldsymbol{z} - \boldsymbol{t}_{\text{off}}\|_p$ is constant for all $\boldsymbol{z} \in \{0,1\}^k$.

To use these objects to reduce (Max-)k-SAT on n variables to a CVPP$_p$ instance with rank n, we must reduce k-SAT to CVP$_p$ with a *fixed* basis matrix $\mathbf{B}_{n,k} \in \mathbb{R}^{d \times n}$. We use the matrix

$$\mathbf{B}_{n,k} := \begin{pmatrix} \Phi_1 \\ \vdots \\ \Phi_M \end{pmatrix}$$

consisting of the on-off-isolating parallelepipeds for each possible k-clause on n variables, stacked on top of each other, where $M := 2^k\binom{n}{k}$. Given a k-SAT formula $\{\phi_{i_1}, \ldots, \phi_{i_m}\}$, we create the target

$$\boldsymbol{t} := \begin{pmatrix} \boldsymbol{t}_1 \\ \vdots \\ \boldsymbol{t}_M \end{pmatrix}$$

such that $\boldsymbol{t}_i = \boldsymbol{t}_{\text{off}}$ if $\phi_i \notin \{\phi_{i_1}, \ldots, \phi_{i_m}\}$ and otherwise $\boldsymbol{t}_i = \boldsymbol{t}^*$. (We are oversimplifying a bit here. In our actual construction, we must shift $\boldsymbol{t}_{\text{off}}$ in a way depending on which literals in the clause are negated. See Section 4.) I.e., we use $\boldsymbol{t}_{\text{off}}$ to "turn off" the clauses that do not appear in our SAT instance.

Finally, we show that (p,k)-on-off-isolating parallelepipeds exist if and only if $(p,k+1)$-isolating parallelepipeds exist. To transform a $(p,k+1)$-isolating parallelepiped $\Phi := (\Phi', \boldsymbol{\phi}_{k+1}), \boldsymbol{t}^*$ into a (p,k)-on-off-isolating parallelepiped, we simply take Φ', $\boldsymbol{t}^*$, and $\boldsymbol{t}_{\text{off}} := \boldsymbol{t} - \boldsymbol{\phi}_{k+1}$. A simple calculation shows that $\|\Phi'\boldsymbol{z} - \boldsymbol{t}_{\text{off}}\|_p = 1$ for all $\boldsymbol{z} \in \{0,1\}^k$ and $\|\Phi'\boldsymbol{z} - \boldsymbol{t}^*\|_p = 1$ for all non-zero $\boldsymbol{z} \in \{0,1\}^k$, as needed.

Hardness of approximation. To prove hardness of approximation, we must show how to reduce *approximate* Max-k-SAT instance with n variables to an approximate CVP$_p$ instance with rank n. The 2^n-hardness of approximate CVP$_p$ described in Theorem 1.1 then follows from the recent Gap-SETH conjecture of Manurangsi [Man19].

The construction shown in Eq. (1) is insufficient to prove hardness of approximation because the presence of the "identity matrix gadget" $2\alpha I_n$ forces the closest vector to be within distance roughly $\alpha n^{1/p}$ to the target. As a result, all SAT instances yield a CVP$_p$ instance with $\text{dist}_p(\boldsymbol{t}, \mathcal{L}) \in (r, (1 + O(1/n))r)$ for some radius $r \approx \alpha n^{1/p}$.

To reduce to approximate CVP$_p$, we therefore need to somehow remove this gadget, which we do by extending isolating parallelepipeds to "isolating lattices." Specifically, we show how to construct a basis $\Phi \in \mathbb{R}^{d^* \times k}$ and target vector $\boldsymbol{t}^* \in \mathbb{R}^{d^*}$ such that $\Phi\boldsymbol{z}$ is a closest lattice vector to $\boldsymbol{t}^*$ if and only if $\boldsymbol{z} \in \{0,1\}^k$ and $\boldsymbol{z}$ corresponds to a satisfying assignment of the k-CNF ϕ. I.e., while previously the satisfying assignments corresponded exactly to the closest vectors to $\boldsymbol{t}^*$ *in the parallelepiped* $\Phi\{0,1\}^k$, now the satisfying assignments must correspond exactly to the closest vectors to $\boldsymbol{t}^*$ *in the entire lattice* $\Phi\mathbb{Z}^k$. This eliminates the need for the identity matrix gadget.

Again, we show how to convert any isolating parallelepiped into a full isolating lattice. The main idea is simply to "append an identity matrix gadget" to the isolating parallelepiped directly, rather than appending it to the full basis as in Eq. (1). Namely, we convert an isolating parallelepiped $\Phi, \boldsymbol{t}^*$ into an isolating lattice $\Phi', \boldsymbol{t}'$ by appending a scaled identity matrix $2\alpha I_k$ to the bottom of Φ, and a constant vector $(\alpha, \alpha, \ldots, \alpha)^T$ to the bottom of $\boldsymbol{t}^*$. By setting α to be large enough, we ensure that any non-binary combination of vectors in Φ' will be far from $\boldsymbol{t}'$. By "putting the identity matrix in

the parallelepiped," rather than in the whole basis, we are able to obtain an approximation factor that depends only on k (and the Gap-k-SAT approximation factor) and not on n.

1.3 Impossibility of natural reductions for $p = 2$

In [BGS17], we showed that the technique described above cannot work for even integers $p < k$. Specifically, we showed that isolating parallelepipeds do not exist in this case. However, this still left open the possibility of some other (potentially even simple) reduction from k-SAT to CVP_p for even integers p—perhaps even for $p = 2$. Here, we show that a very large class of reductions cannot work for $p = 2$. Behind these limitations is a new result concerning the structure of the closest lattice vectors to a target in the Euclidean norm.

Before we define natural reductions and show their limitations, we motivate the definition by showing a simple limitation that applies for all $1 < p < \infty$. Specifically, we recall the well-known fact that for such p, the number of closest lattice vectors to a target is at most $2^{n'}$, where n' is the rank of the lattice. (We show the simple proof of this fact below. Notice that 2^n closest vectors are actually achieved by the integer lattice $\mathcal{L} = \mathbb{Z}^{n'}$ and the all-halves target vector $\boldsymbol{t} = (1/2, \ldots, 1/2)$.) Therefore, if a reduction maps each satisfying assignment of some 3-SAT formula to a distinct closest lattice vector, the rank n of the resulting lattice must be at least $\log_2 S$, where S is the number of satisfying assignments. (Here, and below, we only consider the YES case, when there exists at least one satisfying assignment.) Since the number of satisfying assignments can be as large as 2^n, where n is the number of variables in the input instance, we must have $n' \geq n$.

Our specific reductions described above actually map each assignment $\boldsymbol{z} \in \{0,1\}^n$ to a very simple lattice vector: $\mathbf{B}\boldsymbol{z}$. I.e., $\boldsymbol{z}$ is a satisfying assignment if and only if $\|\mathbf{B}\boldsymbol{z} - \boldsymbol{t}\|_2 = r$. This suggests the following generalization of this type of reduction.

We call a reduction *natural* if there exists a map f from assignments $\boldsymbol{x} \in \{0,1\}^n$ to coordinate vectors $\boldsymbol{z} \in \mathbb{Z}^{n'}$ such that whenever the input 3-SAT formula is satisfiable, $\|\mathbf{B}\boldsymbol{z} - \boldsymbol{t}\|_2 = \mathrm{dist}_2(\boldsymbol{t}, \mathcal{L})$ if and only if $\boldsymbol{z} = f(\boldsymbol{x})$ for some satisfying assignment $\boldsymbol{x} \in \{0,1\}^n$. (We do not require f, or even the reduction itself, to be efficiently computable.) Our reductions described above then correspond to the special case when $n = n'$ and f is the identity map.

Closest vectors mod two. To rule out such reductions for $n' < 4n/3$, we study the algebraic and combinatorial properties of the set $S_{\mathcal{L},\boldsymbol{t}}$ of closest vectors in a lattice $\mathcal{L} \subset \mathbb{R}^n$ to some target vector $\boldsymbol{t}$. To motivate our techniques, let us first recall the well-known simple proof of the fact (mentioned above) that the number of closest vectors is at most $2^{n'}$ for $1 < p < \infty$. Consider two distinct closest vectors $\boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathcal{L}$ to some target $\boldsymbol{t}$. Suppose that $\boldsymbol{y}_1 + \boldsymbol{y}_2 = 2\boldsymbol{v}$ for some lattice vector $\boldsymbol{v} \in \mathcal{L}$. Then, $\|\boldsymbol{v} - \boldsymbol{t}\|_p = \|(\boldsymbol{y}_1 - \boldsymbol{t})/2 + (\boldsymbol{y}_2 - \boldsymbol{t})/2\|_p < \|\boldsymbol{y}_1 - \boldsymbol{t}\|_p/2 + \|\boldsymbol{y}_2 - \boldsymbol{t})\|_p/2$, where we have used the *strict convexity* of the ℓ_p norms for $1 < p < \infty$. (I.e., the triangle inequality $\|\boldsymbol{x} + \boldsymbol{y}\|_p \leq \|\boldsymbol{x}\|_p + \|\boldsymbol{y}\|_p$ is tight for $1 < p < \infty$ if and only if $\boldsymbol{y}$ is a scalar multiple of $\boldsymbol{x}$. Notice that this is false for $p = 1$ and $p = \infty$, and in each of these cases it is easy to show that there can be arbitrarily many closest lattice vectors to a target, even in two dimensions.)

The above proof does not *only* show that the number of closest vectors is at most $2^{n'}$; it also shows that the set $S_{\mathbf{B},\boldsymbol{t}} \subset \mathbb{Z}^{n'}$ of coordinates of closest vectors in some basis $\mathbf{B}$ have some algebraic structure. Specifically, there can be at most one element in $S_{\mathbf{B},\boldsymbol{t}}$ in each *coset* of $\mathbb{Z}^{n'}/(2\mathbb{Z}^{n'})$. Here, a coset is the set $2\mathbb{Z}^{n'} + \boldsymbol{z}$ of all integer vectors with fixed parity. Notice that two cosets can be added together to obtain a new coset, $(2\mathbb{Z}^{n'} + \boldsymbol{z}_1) + (2\mathbb{Z}^{n'} + \boldsymbol{z}_2) = 2\mathbb{Z}^{n'} + (\boldsymbol{z}_1 + \boldsymbol{z}_2)$, and the above proof relied crucially on this structure. Of course, under addition, the cosets are isomorphic to the

vector space of $\mathbb{Z}_2^{n'}$. It is then natural to ask about the structure of $T_{\mathbf{B},\boldsymbol{t}} := S_{\mathbf{B},\boldsymbol{t}} \bmod 2$, viewed as a subset of the hypercube $\mathbb{F}_2^{n'}$.

Indeed, in Section 6 we show the following curious property of $S_{\mathbf{B},\boldsymbol{t}}$ for $p = 2$. Let $C_2 \subset \mathbb{F}_2^{n'}$ be an affine square mod two (i.e., a two-dimensional affine subspace), and suppose that $C_2 \subseteq T_{\mathbf{B},\boldsymbol{t}}$. Let $C \subseteq S_{\mathbf{B},\boldsymbol{t}}$ be the elements such that $C \bmod 2 = C_2$. (As we discussed above, there must be exactly four such elements.) Then, we show that either (1) the points in C form a parallelogram over the reals (i.e., they must have the form $\boldsymbol{z}_1, \boldsymbol{z}_1 + \boldsymbol{z}_2, \boldsymbol{z}_1 + \boldsymbol{z}_3, \boldsymbol{z}_1 + \boldsymbol{z}_2 + \boldsymbol{z}_3$), or (2) there is some specific set of four other elements C' that must also lie in $S_{\mathbf{B},\boldsymbol{t}}$.

Studying the image of f. To see how this can be used to rule out natural reductions, consider the image $A := f(\{0,1\}^n)$ of f and $A_2 := A \bmod 2$. Suppose that A_2 contains an affine square $C_2 \subset A_2$, with corresponding set $C \subset A$. Suppose that C is not a parallelogram over the reals, and let C' be the other four elements guaranteed by the above discussion. Then, let $E := f^{-1}(C) \subset \{0,1\}^n$ and $E' := f^{-1}(C') \subset \{0,1\}^n$ be the corresponding set of assignments. We observe that there exist 3-SAT instances that are satisfied by all elements in E but not all elements in E'. (This can be accomplished with a single clause.) But, our reduction must map any such instance to a basis $\mathbf{B}$ and a target $\boldsymbol{t}$ such that $C', C \subset S_{\mathbf{B},\boldsymbol{t}}$. This contradicts the assumption that f only maps satisfying assignments to closest vectors.

Therefore, whenever A_2 contains an affine square C_2, the corresponding set C in A must be a parallelogram. It follows that any affine *3-cube* in A_2 must correspond to a 3-dimensional parallelepiped P in A. Finally, we find a 3-SAT instance satisfied by exactly seven of the eight elements in $f^{-1}(P)$. It follows that the reduction must produce a parallelepiped with exactly seven out of eight points closest to some target. In [BGS17], we already showed that this is impossible. (We provide a simpler proof in Section 6 as well.)

From this, we conclude that A_2 cannot contain any affine 3-cube.

Using additive combinatorics to finish the proof. Above, we observed that the image A_2 of f modulo 2 cannot contain any 3-cube. But, we have already observed that $|A_2| = 2^n$ (i.e., the closest vectors must be distinct modulo 2). So, $A_2 \subseteq \mathbb{F}_2^{n'}$ is some subset of 2^n points in $\mathbb{F}_2^{n'}$ that contains no affine hypercube. By Szemerédi's cube lemma, we must have $n' \geq 4n/3$, which is what we wished to prove.

In fact, we only need a special case of Szemerédi's cube lemma. We provide a simpler proof of this special case based on the pigeon-hole principle. Though the proof is quite simple, to the authors' knowledge it is novel.

1.4 Related work

The most closely related work to this paper is of course [BGS17]. There are two additional papers showing fine-grained hardness of lattice problems, [AS18a], which showed such results for SVP; and [AC19], which did the same for SIVP. Both of these works relied on the results in [BGS17], and our improvements therefore immediately imply better hardness results for both SVP and SIVP.

An additional line of work has shown different kinds of hardness for CVP, SVP, and related problems. In particular, Bhattacharyya, Ghoshal, Karthik, and Manurangsi showed the parameterized hardness of CVP and SVP, as well as the analogous coding problems [BGKM18]. [SV19] showed tight hardness results for coding problems, using many ideas from [BGS17]. We in turn use some ideas from [SV19], and particular the idea of on-off-isolating parallelepipeds.

Finally, we wish to draw attention to the beautiful Gap-SETH hypothesis of Manurangsi [Man19], presented here in Definition 2.7. The conjecture is quite natural, and we suspect that it will have many additional applications in the study of fine-grained hardness of approximation. E.g., it was already mentioned in [SV19] that something like this Gap-SETH hypothesis would imply strong hardness of approximation results for coding problems.

1.5 Open questions

The most obvious question that we leave open is, of course, to prove similar hardness results for CVP_2, and more generally, for CVP_p for even integers p. In the $p = 2$ case, we show that any such proof (via SETH) would have to use an "unnatural reduction." So, a fundamentally different approach is needed.[2]

Another potentially easier problem would be to show hardness of CVP_p in terms of the ambient dimension d, rather than n. Indeed, though there do exist $2^{O(n)}$-time constant-factor approximation algorithms for CVP_p, the parameter d is in some sense more natural. (E.g., the original algorithm of [BN09] runs in time $2^{O(d)}$, and the algorithm of [AM18] also has its running time in terms of d.) This problem is potentially easier than the above because for $p = 2$ we may assume without loss of generality that $n = d$.

Of course, another open question is to prove stronger quantitative lower bounds for SVP_p, and in particular for SVP_2. While [AS18a] did prove quite strong lower bounds for sufficiently large p, their bounds for small p and in particular for $p = 2$ are quite weak.

We also note that CVP_p for $p \neq 2$ has received relatively little attention from an algorithmic perspective. In particular, there has not been much work trying to optimize the hidden constants in the exponent in the running times of $2^{O(n)}$ or $2^{O(d)}$ of the best known algorithms for constant-factor approximate CVP_p. Our lower bounds provide new motivation for work on this subject. In particular, we ask whether our lower bounds are tight.

In fact, we do not expect our lower bound to be tight in the case when $p = \infty$. (Recall that our limitation in Theorem 1.2 does not apply to $p = 1$ or $p = \infty$.) Indeed, because the kissing number in the ℓ_∞ norm is $3^n - 1$, one might guess that the fastest algorithms for CVP_∞ and SVP_∞ actually run in time $3^{n+o(n)}$ or perhaps $3^{d+o(d)}$. (See [AM18], which more-or-less achieves this.) We therefore ask whether stronger lower bounds can be proven in this special case.

Finally, we note that our results only apply for *exact* CVP_p or CVP_p with a small constant approximation factor. For cryptographic applications, one is interested in much larger approximation factors, typically approximation factors polynomial in n. While there are strong complexity-theoretic barriers to proving hardness in that regime, one might still hope to prove fine-grained hardness results for larger approximation factors—such as large constants or even superconstant. Indeed, we know NP-hardness up to an approximation factor of $n^{c/\log\log n}$, but this result is not fine-grained [DKRS03].

[2] We note that the main reduction in [BGS17] works as a (natural) reduction from weighted Max-2-SAT formulas on n variables with arbitrary (possibly exponential) weights to CVP_p instances of rank n for all $p \in [1, \infty)$, including $p = 2$. So, a $2^{(1-\varepsilon)n}$-time algorithm for CVP_2 would imply a $2^{(1-\varepsilon)n}$-time algorithm for weighted Max-2-SAT with arbitrary weights, for which no such algorithm is known (Ryan Williams' algorithm for Max-2-SAT [Wil05] runs in $O^*(W \cdot 2^{\omega n/3})$-time, where W is the largest weight of a clause). So, there is (potentially weak) evidence that there is no $2^{(1-\varepsilon)n}$-time algorithm for CVP_2.

2 Preliminaries

Throughout this paper, we work with lattice problems over $\mathbb{R}^d$ for convenience. As usual, to be formal we must pick a suitable representation of real numbers and consider both the size of the representation and the efficiency of arithmetic operations in the given representation. But, we omit such details throughout to ease readability.

2.1 Lattice problems

Let $\mathrm{dist}_p(\mathcal{L}, \boldsymbol{t}) := \min_{\boldsymbol{x}\in\mathcal{L}} \|\boldsymbol{x} - \boldsymbol{t}\|_p$ denote the ℓ_p distance of $\boldsymbol{t}$ to $\mathcal{L}$. We next formally define the lattice problems that we consider.

Definition 2.1. *For any $\gamma \geq 1$ and $1 \leq p \leq \infty$, the γ-approximate Shortest Vector Problem with respect to the ℓ_p norm (γ-SVP$_p$) is the promise problem defined as follows. Given a lattice $\mathcal{L}$ (specified by a basis $B \in \mathbb{R}^{d\times n}$) and a number $r > 0$, distinguish between a 'YES' instance where there exists a non-zero vector $\boldsymbol{v} \in \mathcal{L}$ such that $\|\boldsymbol{v}\|_p \leq r$, and a 'NO' instance where $\|\boldsymbol{v}\|_p > \gamma r$ for all non-zero $v \in \mathcal{L}$.*

Definition 2.2. *For any $\gamma \geq 1$ and $1 \leq p \leq \infty$, the γ-approximate Closest Vector Problem with respect to the ℓ_p norm (γ-CVP$_p$) is the promise problem defined as follows. Given a lattice $\mathcal{L}$ (specified by a basis $B \in \mathbb{R}^{d\times n}$), a target vector $\boldsymbol{t} \in \mathbb{R}^d$, and a number $r > 0$, distinguish between a 'YES' instance where $\mathrm{dist}_p(\mathcal{L}, \boldsymbol{t}) \leq r$, and a 'NO' instance where $\mathrm{dist}_p(\mathcal{L}, \boldsymbol{t}) > \gamma r$.*

When $\gamma = 1$, we simply refer to the problems as SVP$_p$ and CVP$_p$.

Definition 2.3. *The Closest Vector Problem with Preprocessing with respect to the ℓ_p norm (*CVPP$_p$*) is the problem of finding a preprocessing function P and an algorithm Q which work as follows. Given a lattice $\mathcal{L}$ (specified by a basis $B \in \mathbb{R}^{d\times n}$), P outputs a new description of $\mathcal{L}$. Given $P(\mathcal{L})$, a target vector $\boldsymbol{t} \in \mathbb{R}^d$, and a number $r > 0$, Q decides whether $\mathrm{dist}_p(\mathcal{L}, \boldsymbol{t}) \leq r$.*

When we measure the runtime of a CVPP algorithm, we only count the runtime of Q, and not of the preprocessing algorithm P. We will assume that the runtime of Q is at least the size of the preprocessing, $|P(L)|$.

2.2 Isolating parallelepipeds

We recall the definition of an *isolating parallelepiped* from [BGS17]. See Figure 1.

Definition 2.4. *For any $1 \leq p \leq \infty$ and integer $k \geq 1$, we say that $V \in \mathbb{R}^{d^*\times k}$ and $\boldsymbol{t}^* \in \mathbb{R}^{d^*}$ define a* (p, k)-isolating parallelepiped *if:*

1. $\|V\boldsymbol{x} - \boldsymbol{t}^*\|_p = 1$ *for all* $\boldsymbol{x} \in \{0, 1\}^k \setminus \{\boldsymbol{0}\}$,
2. $\|\boldsymbol{t}^*\|_p > 1$.

We will more generally refer to the set $V \cdot \{0, 1\}^k - \boldsymbol{t}^*$ for $V \in \mathbb{R}^{d^*\times k}$ and $\boldsymbol{t}^* \in \mathbb{R}^{d^*}$ as a *k-parallelepiped*. We call a 2-parallelepiped a *parallelogram*.

2.3 k-SAT

A *k-SAT* formula Φ on n boolean variables $x_1, \ldots, x_n$ is the conjunction $\wedge_{i=1}^{m} C_i$ of m clauses, each of which is a dijunction $C_i = \vee_{s=1}^{k} \ell_{i,k}$ of k literals. Each literal $\ell_{i,k}$ is either a variable x_j or its negation $\neg x_j$ for some $j \in [n]$. The k-SAT problem is, given a k-SAT formula Φ, to decide whether there exists an assignment $\boldsymbol{a}$ to the variables of Φ that satisfies Φ, i.e., such that $\Phi(\boldsymbol{a}) = 1$.

We next introduce some notation related to k-SAT. Let Φ be a k-SAT formula on n variables $x_1, \ldots, x_n$ and m clauses $C_1, \ldots, C_m$. Let $\mathrm{ind}(\ell)$ denote the index of the variable underlying a literal ℓ. I.e., $\mathrm{ind}(\ell) = j$ if $\ell = x_j$ or $\ell = \neg x_j$. Call a literal ℓ *positive* if $\ell = x_j$ and *negative* if $\ell = \neg x_j$ for some variable x_j. Given a clause $C_i = \vee_{s=1}^{k} \ell_{i,s}$, let $P_i := \{s \in [k] : \ell_{i,s} \text{ is positive}\}$ and let $N_i := \{s \in [k] : \ell_{i,s} \text{ is negative}\}$ denote the indices of positive and negative literals in C_i respectively. Given an assignment $\boldsymbol{a} \in \{0,1\}^n$ to the variables of Φ, let $S_i(\boldsymbol{a})$ denote the indices of literals in C_i satisfied by $\boldsymbol{a}$. I.e., $S_i(\boldsymbol{a}) := \{s \in P_i : a_{\mathrm{ind}(\ell_{i,s})} = 1\} \cup \{s \in N_i : a_{\mathrm{ind}(\ell_{i,s})} = 0\}$. Finally, when a formula Φ is clear from context, let $m^+(\boldsymbol{a})$ denote the number of clauses of Φ satisfied by the assignment $\boldsymbol{a}$, i.e., the number of clauses i for which $|S_i(\boldsymbol{a})| \geq 1$.

The *value* of a k-SAT formula Φ, denoted $\mathrm{val}(\Phi)$, is the maximum fraction of clauses satisfied by an assignment to Φ.

Definition 2.5. *Given a k-SAT formula Φ and constants $0 \leq \delta \leq \varepsilon \leq 1$, the (δ, ε)-Gap-k-SAT problem is the promise problem defined as follows. The goal is to distinguish between a 'YES' instance in which $\mathrm{val}(\Phi) \geq \varepsilon$, and a 'NO' instance in which $\mathrm{val}(\Phi) < \delta$.*

2.4 Hardness assumptions

Definition 2.6 (SETH; [IPZ01])**.** *For every $\varepsilon > 0$ there exists a $k = k(\varepsilon) \in \mathbb{Z}^+$ such that no algorithm solves k-SAT on n variables in $2^{(1-\varepsilon)n}$ time.*

In his Ph.D. thesis, Manurangsi [Man19] gave one possible definition of Gap-SETH.

Definition 2.7 (Gap-SETH; [Man19, Conjecture 12.1])**.** *For every $\varepsilon > 0$ there exist $k = k(\varepsilon) \in \mathbb{Z}^+$ and $\delta = \delta(\varepsilon) > 0$ such that there is no algorithm that can distinguish between a k-SAT formula with n variables that is satisfiable and one that has value less than $1 - \delta$ in $2^{(1-\varepsilon)n}$ time.*

We will show that CVP_p cannot be approximated to within some factor $\gamma_\varepsilon > 1$ in $2^{(1-\varepsilon)n}$ time assuming Gap-SETH. Unfortunately, γ_ε decays as a function of ε. However, our reduction from Gap-k-SAT to CVP_p can be adapted to a reduction from *any* Gap-k-CSP to CVP_p with the same relevant parameters. (Namely, our reduction maps CSP instances on n variables to CVP(P) instances of rank n.)

We will also use non-uniform variants of ETH and SETH to prove hardness results about CVPP_p.

Definition 2.8 (Non-uniform ETH)**.** *There is no family of circuits of size $2^{o(n)}$ that solves 3-SAT instances on n variables.*

Definition 2.9 (Non-uniform SETH)**.** *For every $\varepsilon > 0$ there exists a $k = k(\varepsilon) \in \mathbb{Z}^+$ such that no family of circuits of size $2^{(1-\varepsilon)n}$ solves k-SAT instances on n variables.*

Our results are also quite robust to how we define non-uniform (S)ETH. For example, one of our main results about the complexity of CVPP_p roughly says that assuming non-uniform ETH (as stated above) there is no subexponential-sized family of circuits that decides CVPP_p for $p \neq 2$.

However, if we were to change non-uniform ETH to say that there is no $2^{o(n)}$-time algorithm using $\mathrm{poly}(n)$ advice, then we would get a corresponding statement for CVPP_p: that there is no $2^{o(n)}$-time algorithm for CVPP_p using $\mathrm{poly}(n)$ advice.

Interestingly, many of our results only depend on weaker versions of these hypotheses, where we replace an assumption about the hardness of k-SAT with an assumption about the hardness of Max-k-SAT or even weighted Max-k-SAT.

2.5 Linear algebra

We recall that an *affine k-cube* in $\mathbb{F}_2^n$ is $\{\boldsymbol{y}_0 + \sum_{j \in W} \boldsymbol{y}_j \ : \ W \subseteq \{1, \ldots, k\}\}$ for some $\boldsymbol{y}_0 \in \mathbb{F}_2^n$ and linearly independent $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k \in \mathbb{F}_2^n$.

We will use the following determinant identity for block matrices.

Fact 2.10. *Let $A, B \in \mathbb{R}^{n \times n}$ for some $n \in \mathbb{Z}^+$. Then*

$$\det \begin{pmatrix} A & B \\ B & A \end{pmatrix} = \det(A - B) \cdot \det(A + B) \ .$$

We say that functions $f_0, \ldots, f_n : \mathbb{R} \to \mathbb{R}$ are linearly independent over the reals if given $a_0, \ldots, a_n \in \mathbb{R}$, the sum $\sum_{i=0}^n a_i f_i(x)$ is identically zero (is equal to 0 for all $x \in \mathbb{R}$) only if $a_0 = \cdots = a_n = 0$. We say that $f \in C^k$ if the first k derivatives of f exist and are continuous, $f \in C^\infty$ if f has derivatives of all orders, and that f is *analytic* if $f \in C^\infty$ and if the Taylor series of f expanded around any point x in the domain converges to f in some neighborhood of x. We say that $f \in C^k(a, b)$ if the first k derivatives of f exist and are continuous on the (open) interval (a, b) (we define $f \in C^\infty(a, b)$ and f being analytic on (a, b) analogously).

Definition 2.11. *We define the* Wronskian *of $f_0, \ldots, f_n \in C^n(a, b)$ to be $\det(M)$, where M is the $(n+1) \times (n+1)$ matrix defined by*

$$M := \begin{pmatrix} f_0(x) & f_1(x) & \cdots & f_n(x) \\ \frac{d}{dx} f_0(x) & \frac{d}{dx} f_1(x) & \cdots & \frac{d}{dx} f_n(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{d^n}{dx^n} f_0(x) & \frac{d^n}{dx^n} f_1(x) & \cdots & \frac{d^n}{dx^n} f_n(x) \end{pmatrix}$$

for $x \in (a, b)$.

Because the derivative is a linear operator, we have the following.

Fact 2.12. *Functions $f_0, \ldots, f_n$ are linearly independent over the reals if their Wronskian exists and is not identically zero on some interval (a, b).*

3 Isolating parallelepipeds in ℓ_p norms for all non-integer p

Our first new result is a strengthening of a result in [BGS17], which asserts that for every fixed $k \in \mathbb{Z}^+$ there exist (p, k)-isolating parallelepipeds for *almost every* $p \in [1, \infty) \setminus 2\mathbb{Z}$, to a result showing that this is true for *every* $p \in [1, \infty) \setminus 2\mathbb{Z}$. We also show that there exist (p, k)-isolating

parallelepipeds when $k \leq p$. Moreover, we show that one of these conditions is also necessary, and therefore obtain a complete characterization of values of p and k for which isolating parallelepipeds exist (such isolating parellelepipeds are computable if p is computable).

Our construction generalizes the approach from [BGS17], and follows the same high-level structure. We start by showing that it suffices to "define isolating parallelepipeds over $\{-1,1\}$ instead of $\{0,1\}$," i.e., that if there exist $V = (\boldsymbol{v}_1, \ldots, v_k) \in \mathbb{R}^{d\times k}$ and $\boldsymbol{t}^* \in \mathbb{R}^d$ that satisfy $\|V\boldsymbol{y} - \boldsymbol{t}^*\|_p = 1$ for $\boldsymbol{y} \in \{-1,1\}^k \setminus \{-\mathbf{1}\}$ and $\|V(-\mathbf{1}) - \boldsymbol{t}^*\|_p > 1$, then there exists a (p,k)-isolating parallelepiped.

We then define a family of k-parallelepipeds $V \in \mathbb{R}^{2^k\times k}, \boldsymbol{t}^* \in \mathbb{R}^{2^k}$ parameterized by 2^k numbers, $\alpha_{\boldsymbol{u}} \geq 0$ for $\boldsymbol{u} \in \{-1,1\}^k$, and a number t^*. Specifically, the row of V indexed by $\boldsymbol{u} \in \{-1,1\}^k$ is equal to $\alpha_{\boldsymbol{u}}^{1/p} \cdot \boldsymbol{u}^T$ and the coordinate of $(\boldsymbol{t}^*)_{\boldsymbol{u}} = \alpha_{\boldsymbol{u}}^{1/p} \cdot t^*$. (Throughout this section, we will adopt the convention that vectors $\boldsymbol{v} \in \mathbb{R}^{2^k}$ for some $k \in \mathbb{Z}^+$ are indexed by elements in $\{-1,1\}^k$ in lexicographic order. We adopt an analogous convention for rows (resp. columns) of matrices of the form $M \in \mathbb{R}^{2^k\times m}$ (resp. $M \in \mathbb{R}^{m\times 2^k}$) for some m.) Figure 2 shows the form of such a k-parallelepiped when $k = 3$.

We observe that for such a family of k-parallelepipeds and $\boldsymbol{y} \in \{-1,1\}^k$, $\|V\boldsymbol{y} - \boldsymbol{t}^*\|_p^p = \sum_{\boldsymbol{u}} \alpha_{\boldsymbol{u}} |\langle \boldsymbol{u}, \boldsymbol{y}\rangle - t^*\rangle|^p$. I.e., for fixed $\boldsymbol{y}$ and $\boldsymbol{t}^*$, $\|V\boldsymbol{y} - \boldsymbol{t}^*\|_p^p$ is linear in the values $\alpha_{\boldsymbol{u}}$. This leads us to define the $2^k \times 2^k$ matrix $H_{k,p}(t^*)$ whose entry in row $\boldsymbol{u}$ and column $\boldsymbol{y}$ is equal to $|\langle \boldsymbol{u}, \boldsymbol{y}\rangle - t^*|^p$. Then, for non-negative $\boldsymbol{\alpha} = (\alpha_{\boldsymbol{u}})_{\boldsymbol{u}\in\{-1,1\}^k}$, the coordinate of $H_{k,p}(t^*) \cdot \boldsymbol{\alpha}$ indexed by $\boldsymbol{y}$ is equal to $\|V\boldsymbol{y} - \boldsymbol{t}^*\|_p^p$.

In order to show that there exist choices of $\boldsymbol{\alpha}$ and t^* such that V and $\boldsymbol{t}^*$ form a "$\{-1,1\}$ isolating parallelepiped," it therefore suffices to find non-negative $\boldsymbol{\alpha}$ such that $H_{k,p}(t^*)\cdot\boldsymbol{\alpha} = (1+\varepsilon, 1, 1, \ldots, 1)^T$ for some $\varepsilon > 0$. We then use the following proof strategy for finding such $\boldsymbol{\alpha}$: (1) Show that for certain values of k and p, $H_{k,p}(t^*)$ is non-singular so that we can compute $\boldsymbol{\alpha} = H_{k,p}(t^*)^{-1}\cdot(1+\varepsilon, 1, 1, \ldots, 1)^T$, and (2) show that if we pick $\varepsilon > 0$ to be small enough then $\boldsymbol{\alpha}$ computed this way will be non-negative.

3.1 A parameterized family of parallelepipeds

We recall the following lemma from [BGS17], which says that we can "work over $\{-1,1\}$ instead of $\{0,1\}$" when defining isolating parallelepipeds, which we will do in this section. We include its short proof for completeness.

Lemma 3.1 ([BGS17, Lemma 4.1])**.** *There is an efficient algorithm that takes as input a matrix $V \in \mathbb{R}^{d^*\times k}$ and vector $\boldsymbol{t}^* \in \mathbb{R}^{d^*}$ such that $\|V\boldsymbol{y} - \boldsymbol{t}^*\|_p = 1$ for any $\boldsymbol{y} \in \{-1,1\}^k \setminus \{-\mathbf{1}\}$ and $\|-V\mathbf{1} - \boldsymbol{t}^*\|_p > 1$, and outputs a matrix $V' \in \mathbb{R}^{d^*\times k}$ and vector $(\boldsymbol{t}^*)' \in \mathbb{R}^{d^*}$ that form a (p,k)-isolating parallelepiped.*

Proof. Define $V' := 2V$ and $(\boldsymbol{t}^*)' = V\mathbf{1}_k + \boldsymbol{t}^*$. Now consider the affine transformation $f\colon \mathbb{R}^k \to \mathbb{R}^k$ defined by $f(\boldsymbol{x}) := (2\boldsymbol{x} - \mathbf{1}_k)$, which maps $\{0,1\}^k$ to $\{\pm 1\}^k$ and $\mathbf{0}$ to $-\mathbf{1}$. Then, for $\boldsymbol{x} \in \{0,1\}^k$ and $\boldsymbol{y} = f(\boldsymbol{x}) = 2\boldsymbol{x} - \mathbf{1} \in \{\pm 1\}^k$, we have

$$\|V'\boldsymbol{x} - (\boldsymbol{t}^*)'\|_p = \left\|V'\frac{\boldsymbol{y}+\mathbf{1}}{2} - (\boldsymbol{t}^*)'\right\|_p = \left\|V'\frac{\boldsymbol{y}}{2} + V'\frac{\mathbf{1}}{2} - (\boldsymbol{t}^*)'\right\|_p = \|V\boldsymbol{y} - \boldsymbol{t}^*\|_p\,,$$

as needed. □

We next define a family of k-parallelepipeds $V \in \mathbb{R}^{2^k\times k}, \boldsymbol{t}^* \in \mathbb{R}^{2^k}$ parameterized by 2^k nonnegative numbers $\{\alpha_{\boldsymbol{u}}\}_{\boldsymbol{u}\in\{-1,1\}^k}$, where, for some $p \geq 1$, $\alpha_{\boldsymbol{u}}^{1/p}$ scales the row of V and coordinate of $\boldsymbol{t}^*$ corresponding to $\boldsymbol{u} \in \{-1,1\}^k$, and a number t^*.

$$V := \begin{pmatrix} \alpha_{(-1,-1,-1)}^{1/p} & 0 & \cdots & 0 \\ 0 & \alpha_{(-1,-1,1)}^{1/p} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_{(1,1,1)}^{1/p} \end{pmatrix} \cdot \begin{pmatrix} -1 & -1 & -1 \\ -1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & -1 \\ -1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\boldsymbol{t}^* := \begin{pmatrix} \alpha_{(-1,-1,-1)}^{1/p} & 0 & \cdots & 0 \\ 0 & \alpha_{(-1,-1,1)}^{1/p} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_{(1,1,1)}^{1/p} \end{pmatrix} \cdot \begin{pmatrix} t^* \\ t^* \\ t^* \\ t^* \\ t^* \\ t^* \\ t^* \\ t^* \end{pmatrix}$$

Figure 2: V and $\boldsymbol{t}^*$ of the form defined in Definition 3.2 for $k = 3$ and $p \geq 1$. Lemma 3.1 and Lemma 3.9 together assert that, for $p \in [1, \infty)$ where p satisfies either (1) $p \notin \mathbb{Z}$ or (2) $p \geq 3$, there exist $\{\alpha_{\boldsymbol{u}}\}_{\boldsymbol{u} \in \{-1,1\}^3}$ and t^* such that $V' := 2V$, $(\boldsymbol{t}^*)' := V\mathbf{1} + \boldsymbol{t}^*$ form an isolating parallelepiped.

Definition 3.2. *For $p \in [1, \infty)$, $k \in \mathbb{Z}^+$, $\boldsymbol{\alpha} \in (\mathbb{R}^{\geq 0})^{2^k}$, and $t^* \in \mathbb{R}$, define the matrix $V = V(\boldsymbol{\alpha}) \in \mathbb{R}^{2^k \times k}$ and vector $\boldsymbol{t}^* = \boldsymbol{t}^*(\boldsymbol{\alpha}, t^*) \in \mathbb{R}^{2^k}$ as follows. Set the row of V indexed by $\boldsymbol{u}$ to be $\alpha_{\boldsymbol{u}}^{1/p} \cdot \boldsymbol{u}^T$, and set $\boldsymbol{t}^* := t^* \cdot (\alpha_{\boldsymbol{u}}^{1/p})_{\boldsymbol{u} \in \{-1,1\}^k}$.*

I.e., V is the matrix whose rows consist of vectors $\boldsymbol{u} \in \{-1, 1\}^k$ scaled by corresponding weights $\alpha_{\boldsymbol{u}}^{1/p}$, and the coordinate of $\boldsymbol{t}^*$ indexed by $\boldsymbol{u}$ is equal to $\alpha_{\boldsymbol{u}}^{1/p} \cdot t^*$. (See Figure 2.) We also define another matrix, H, which we will use to relate our choice of parameters $\boldsymbol{\alpha}$ and t^* to the value of $\|V\boldsymbol{y} - \boldsymbol{t}^*\|_p^p$ for $\boldsymbol{y} \in \{-1, 1\}^k$.

Definition 3.3. *For $p \geq 1$ and an integer $k \geq 0$, define the matrix $H_{k,p}(t^*) \in \mathbb{R}^{2^k \times 2^k}$ by $(H_{k,p}(t^*))_{\boldsymbol{u},\boldsymbol{v}} := |\langle \boldsymbol{u}, \boldsymbol{v} \rangle - t^*|^p$ for $k \geq 1$, and define $H_{0,p}(t^*) := |t^*|^p$.*

We next show that for $\boldsymbol{y} \in \{-1, 1\}^k$, $\|V\boldsymbol{y} - \boldsymbol{t}^*\|_p^p$ is equal to the inner product of $\boldsymbol{\alpha}$ with row $\boldsymbol{y}$ of $H_{k,p}(t^*)$.

Lemma 3.4. *For $\boldsymbol{\alpha} \in (\mathbb{R}^{\geq 0})^{2^k}$ and $t^* \in \mathbb{R}$, let $V = V(\boldsymbol{\alpha})$ and let $\boldsymbol{t}^* = \boldsymbol{t}^*(\boldsymbol{\alpha}, t^*)$ be as defined in Definition 3.2. Then*

$$(H_{k,p}(t^*) \cdot \boldsymbol{\alpha})_{\boldsymbol{y}} = \|V\boldsymbol{y} - \boldsymbol{t}^*\|_p^p \ .$$

Proof. For $\boldsymbol{y} \in \{-1,1\}^k$,

$$\begin{aligned}(H_{k,p}(t^*) \cdot \boldsymbol{\alpha})_{\boldsymbol{y}} &= \sum_{\boldsymbol{u} \in \{-1,1\}^k} H_{k,p}(t^*)_{\boldsymbol{y},\boldsymbol{u}} \cdot \alpha_{\boldsymbol{u}} \\ &= \sum_{\boldsymbol{u} \in \{-1,1\}^k} |\langle \alpha_{\boldsymbol{u}}^{1/p} \cdot \boldsymbol{u}, \boldsymbol{y} \rangle - \alpha_{\boldsymbol{u}}^{1/p} \cdot t^*|^p \\ &= \|V\boldsymbol{y} - \boldsymbol{t}^*\|_p^p \ ,\end{aligned}$$

as needed. □

3.2 Non-singularity of H with certain parameters

We next show that for every $k \in \mathbb{Z}^+$ and every $p \in [1,\infty)$ that satisfies either (1) $p \notin \mathbb{Z}$ or (2) $p \geq k$, there exists $t^* \in \mathbb{R}$ such that $H_{k,p}(t^*)$ is non-singular. To show non-singularity, we start with a general structural result about $\det(H_{k,p}(t^*))$ as a function of t^*.

Lemma 3.5. *Let $k \geq 0$ be an integer. Then $\det(H_{k,p}(t^*))$ is equal to the product of 2^k functions of the form*

$$\sum_{j=0}^{k} a_j \cdot |t^* - k + 2j|^p \tag{2}$$

for some $a_0, a_1, \ldots, a_k \in \mathbb{Z}$ with $a_0 = 1$.

Proof. We prove the lemma by induction on k, using the following strengthened induction hypothesis.

Induction hypothesis: For every $k \geq 0$, $m \geq 0$, and $a_0, a_1, \ldots, a_m \in \mathbb{Z}$ with $a_0 = 1$,

$$\det\Big(\sum_{j=0}^{m} a_j \cdot H_{k,p}(t^* + 2j)\Big)$$

is equal to the product of 2^k functions of the form

$$\sum_{j=0}^{m+k} b_j \cdot |t^* - k + 2j|^p \ ,$$

with $b_0, b_1, \ldots, b_m \in \mathbb{Z}$ and $b_0 = 1$.

Base case: In the base case where $k = 0$, we have by definition that $\det(\sum_{j=0}^{m} a_j \cdot H_{k,p}(t^* + 2j)) = \sum_{j=0}^{m} a_j \cdot |-2j - t^*|^p = \sum_{j=0}^{m} a_j \cdot |t^* + 2j|^p$, as needed.

Inductive case: We next consider the case where $k \geq 1$. Let $\boldsymbol{u} = (u_1, \boldsymbol{u}')^T, \boldsymbol{v} = (v_1, \boldsymbol{v}')^T \in \{-1,1\}^k$. If $u_1 = v_1$ we then have that $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \langle \boldsymbol{u}', \boldsymbol{v}' \rangle + 1$, and if $u_1 \neq v_1$ then $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \langle \boldsymbol{u}', \boldsymbol{v}' \rangle - 1$ (with $\langle \boldsymbol{u}', \boldsymbol{v}' \rangle = 0$ if $\boldsymbol{u}', \boldsymbol{v}'$ are of length 0). Therefore, we can write $H_{k,p}(t^*)$ in block form as

$$H_{k,p}(t^*) = \begin{pmatrix} H_{k-1,p}(t^* - 1) & H_{k-1,p}(t^* + 1) \\ H_{k-1,p}(t^* + 1) & H_{k-1,p}(t^* - 1) \end{pmatrix} \ ,$$

and so

$$\sum_{j=0}^{m} a_j \cdot H_{k,p}(t^* + 2j) = \begin{pmatrix} \sum_{j=0}^{m} a_j \cdot H_{k-1,p}(t^* + 2j - 1) & \sum_{j=0}^{m} a_j \cdot H_{k-1,p}(t^* + 2j + 1) \\ \sum_{j=0}^{m} a_j \cdot H_{k-1,p}(t^* + 2j + 1) & \sum_{j=0}^{m} a_j \cdot H_{k-1,p}(t^* + 2j - 1) \end{pmatrix} .$$

We can therefore apply the block matrix determinant formula from Fact 2.10 to obtain

$$\begin{aligned} \det\Big(\sum_{j=0}^{m} a_j \cdot H_{k,p}(t^* + 2j)\Big) = \det\Big(\sum_{j=0}^{m} a_j \cdot H_{k-1,p}(t^* + 2j - 1) - \sum_{j=0}^{m} a_j \cdot H_{k-1,p}(t^* + 2j + 1)\Big) \\ \cdot \det\Big(\sum_{j=0}^{m} a_j \cdot H_{k-1,p}(t^* + 2j - 1) + \sum_{j=0}^{m} a_j \cdot H_{k-1,p}(t^* + 2j + 1)\Big) . \end{aligned}$$

Each of the two terms in the product in the above expression is then of the form

$$\det\Big(\sum_{j=0}^{m+1} a'_j \cdot H_{k-1,p}((t^* - 1) + 2j)\Big) \tag{3}$$

for some $a'_0, a'_1, \ldots, a'_{m+1} \in \mathbb{Z}$ with $a'_0 = 1$ (since $a_0 = 1$ by the induction hypothesis). Moreover, by the induction hypothesis the expression in Equation (3) is equal to the product of 2^{k-1} functions of the form

$$\sum_{j=0}^{(m+1)+(k-1)} b_j \cdot |(t^* - 1) + 2j - (k-1)|^p = \sum_{j=0}^{m+k} b_j \cdot |t^* - k + 2j|^p$$

with $b_0 = 1$. It follows that $\det(\sum_{j=0}^{m} a_j \cdot H_{k,p}(t^* + 2j))$ is equal to the product of 2^k functions of this form, as needed.

□

We next show that the function $t^* \mapsto \det(H_{k,p}(t^*))$ is analytic and not identically zero for certain k and p. Using the general fact that such functions have isolated roots, this leads to a simple algorithm for finding t^* such that $\det(H_{k,p}(t^*))$ is non-singular for such k and p.

Proposition 3.6. *Let $k \in \mathbb{Z}^+$, and let $p \in [1, \infty)$ be a value that satisfies either (1) $p \notin \mathbb{Z}$ or (2) $p \geq k$. Then $\det(H_{k,p}(t^*))$ is analytic and not identically zero as a function of t^* for $t^* > k$.*

Proof. We note that functions of the form $|t^* - k + 2j|^p$ for $j \in \{0, 1, \ldots, k\}$ satisfy $|t^* - k + 2j|^p = (t^* - k + 2j)^p$ and are analytic for $t^* > k$. By Lemma 3.5, $\det(H_{k,p}(t^*))$ is a product of 2^k linear combinations of functions of this form (as in Equation (2)). This implies that $\det(H_{k,p}(t^*))$ is also analytic for $t^* > k$, and moreover that in order to show that $\det(H_{k,p}(t^*))$ is not identically zero it suffices to show that each of these linear combination is not identically zero.

Lemma 3.5 further asserts that each of the linear combinations $\sum_{j=0}^{k} a_j \cdot |t^* - k + 2j|^p$ appearing as terms in the expansion of $\det(H_{k,p}(t^*))$ has $a_0 = 1$, and in particular that it is not the all-zeros combination. So, to show that $\sum_{j=0}^{k} a_j \cdot |t^* - k + 2j|^p$ is not identically zero, it suffices to show that the functions $|t^* - k + 2j|^p$ for $j \in \{0, 1, \ldots, k\}$ are linearly independent over the reals. Moreover, it suffices to show that these functions are linearly independent for $t^* > k$, and therefore to show that the functions $(t^* - k + 2j)^p$ for each j are linearly independent, since $|t^* - k + 2j|^p = (t^* - k + 2j)^p$ for $t^* > k$.

By Fact 2.12, to show that these functions are linearly independent it suffices to show that their Wronskian $W := \det(M)$ with

$$M = M_{p,k}(t^*) := \begin{pmatrix} (p)_0 \cdot (t^*-k)^p & (p)_0 \cdot (t^*-k+2)^p & \cdots & (p)_0 \cdot (t^*+k)^p \\ (p)_1 \cdot (t^*-k)^{p-1} & (p)_1 \cdot (t^*-k+2)^{p-1} & \cdots & (p)_1 \cdot (t^*+k)^{p-1} \\ \vdots & \vdots & \ddots & \vdots \\ (p)_k \cdot (t^*-k)^{p-k} & (p)_k \cdot (t^*-k+2)^{p-k} & \cdots & (p)_k \cdot (t^*+k)^{p-k} \end{pmatrix}$$

is not identically zero for $t^* > k$. Here the notation $(p)_i$ denotes the falling factorial function, which is defined by $(p)_i := p(p-1)\cdots(p-(i-1))$ for $i \geq 1$ and $(p)_0 := 1$.

We note that W is not identically zero if and only if the determinant of M with its rows or columns multiplied by a non-zero function of t^* is not identically zero. Accordingly, dividing the ith row of M (which has rows indexed by $i \in \{0, 1, \ldots, k\}$) by $(p)_i$ (which is non-zero because of our assumptions about p) we obtain

$$M' = M'_{p,k}(t^*) := \begin{pmatrix} (t^*-k)^p & (t^*-k+2)^p & \cdots & (t^*+k)^p \\ (t^*-k)^{p-1} & (t^*-k+2)^{p-1} & \cdots & (t^*+k)^{p-1} \\ \vdots & \vdots & \ddots & \vdots \\ (t^*-k)^{p-k} & (t^*-k+2)^{p-k} & \cdots & (t^*+k)^{p-k} \end{pmatrix}.$$

Similarly, dividing the jth column of M' (which has columns indexed by $j \in \{0, 1, \ldots, k\}$) by $(t^*-k+2j)^{p-k}$ (which is well-defined and non-zero for $t^* > k$) we obtain

$$M'' = M''_{p,k}(t^*) := \begin{pmatrix} (t^*-k)^k & (t^*-k+2)^k & \cdots & (t^*+k)^k \\ (t^*-k)^{k-1} & (t^*-k+2)^{k-1} & \cdots & (t^*+k)^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix},$$

which is a Vandermonde matrix up to transposition and reordering of the rows. We can therefore use the formula for the determinant of a Vandermonde matrix to compute

$$\det(M'') = -\prod_{0 \leq i < j \leq k} ((t^*-k+2j) - (t^*-k+2i)) = -\prod_{0 \leq i < j \leq k} 2(j-i) \neq 0$$

Hence W is not identically zero, as needed. □

Corollary 3.7. *For every $k \in \mathbb{Z}^+$ and every real $p \in [1, \infty)$ that satisfies either (1) $p \notin \mathbb{Z}$ or (2) $p \geq k$, there exists t^* such that $\det(H_{k,p}(t^*)) \neq 0$. Moreover, if p is computable then there is an algorithm that on input k and p outputs such a t^*.*

Proof. The corollary is an immediate consequence of Proposition 3.6 and the fact that an analytic function that is not identically zero has isolated roots. Indeed, the fact that such a function has isolated roots implies that the following algorithm must halt (when p is computable). Compute $\det(H_{k,p}(t_i^*))$ where $t_i^* = k + 2^{-i}$ for $i = 1, 2, \ldots$, and output the first t_i^* for which $\det(H_{k,p}(t_i^*)) \neq 0$. □

3.3 Finishing the proof

We next prove that the matrix $H_{k,p}(t^*)$ is *stochastic*, i.e., that it has $\mathbf{1} = (1, 1, \ldots, 1)^T$ as an eigenvector. This essentially follows from the fact that for arbitrary $\boldsymbol{u}_1, \boldsymbol{u}_2 \in \{-1,1\}^k$ and uniformly random $\boldsymbol{v} \sim \{-1,1\}^k$, the distributions $\langle \boldsymbol{u}_1, \boldsymbol{v}\rangle$ and $\langle \boldsymbol{u}_2, \boldsymbol{v}\rangle$ are identical, which in turn follows from the fact that the value of $\langle \boldsymbol{u}, \boldsymbol{v}\rangle$ is determined solely by the number of coordinates on which $\boldsymbol{u}, \boldsymbol{v} \in \{-1,1\}^k$ agree.

Lemma 3.8. *For every $k \in \mathbb{Z}^+$ and every $p \in [1,\infty)$, $H_{k,p}(t^*)$ has $\mathbf{1}$ as an eigenvector with corresponding eigenvalue*

$$\lambda = \sum_{j=0}^{k} \binom{k}{j} |t^* - k + 2j|^p > 0 \ .$$

Proof. Fix a row index $\boldsymbol{u} \in \{-1,1\}^k$. Then

$$(H_{k,p}(t^*) \cdot \mathbf{1})_{\boldsymbol{u}} = \sum_{\boldsymbol{v} \in \{-1,1\}^k} |\langle \boldsymbol{u}, \boldsymbol{v}\rangle - t^*|^p = \sum_{j=0}^{k} \binom{k}{j} |k - 2j - t^*|^p = \sum_{j=0}^{k} \binom{k}{j} |t^* - k + 2j|^p \ ,$$

where we have used the fact that $\langle \boldsymbol{u}, \boldsymbol{v}\rangle = k - 2j$ if and only if $\boldsymbol{u}_i \neq \boldsymbol{v}_i$ for exactly j coordinates $i \in \{1, \ldots, k\}$. Because $\sum_{\boldsymbol{v} \in \{-1,1\}^k} H_{k,p}(t^*)_{\boldsymbol{u},\boldsymbol{v}}$ does not depend on $\boldsymbol{u}$, we get that $(1, 1, \ldots, 1)^T$ is an eigenvector of $H_{k,p}(t^*)$ with corresponding eigenvalue $\sum_{j=0}^{k} \binom{k}{j} |t^* - k + 2j|^p$. Each term in this sum is non-negative, and at most one term is zero. By the assumption that $k \geq 1$, the sum has at least two terms and is therefore positive, as claimed. □

If H is non-singular then for any vector $\boldsymbol{b}$ we can solve the linear system $H_{k,p}(t^*) \cdot \boldsymbol{\alpha} = \boldsymbol{b}$ to obtain some solution $\boldsymbol{\alpha}$. In particular, we can set $\boldsymbol{b} = (1 + \varepsilon, 1, 1, \ldots, 1)$ for some $\varepsilon > 0$ and then solve for $\boldsymbol{\alpha}$. The issue with this is that we critically require that our solution $\boldsymbol{\alpha}$ be non-negative, and a priori there is no guarantee that it will be. However, we next show that by setting $\varepsilon > 0$ to be sufficiently small we can ensure $\boldsymbol{\alpha}$ will in fact be non-negative.

Lemma 3.9. *Let $k \in \mathbb{Z}^+$, let $p \in [1,\infty)$ be a number that satisfies either (1) $p \notin \mathbb{Z}$ or (2) $p \geq k$. Then there exists a vector $\boldsymbol{\alpha} \in (\mathbb{R}^{\geq 0})^{2^k}$ with the property that*

$$H_{k,p}(t^*) \cdot \boldsymbol{\alpha} = \mathbf{1} + \varepsilon \boldsymbol{e}_1 = (1 + \varepsilon, 1, 1, \ldots, 1)$$

for some $\varepsilon > 0$. Moreover, there is an algorithm that, on input $k \in \mathbb{Z}^+$ and any computable $p \in [1,\infty)$ with either (1) $p \notin \mathbb{Z}$ or (2) $p \geq k$, outputs such a vector $\boldsymbol{\alpha}$.

Proof. By Corollary 3.7, there exists $t^* > k$ such that $H_{k,p}(t^*)$ is non-singular. Fix such an t^*, and let $\boldsymbol{\alpha}' := H_{k,p}(t^*)^{-1} \cdot \boldsymbol{e}_1$. By Lemma 3.8, $\mathbf{1}$ is an eigenvector of $H_{k,p}(t^*)$ with corresponding eigenvalue $\lambda = \sum_{j=0}^{k} \binom{k}{j} |t^* - k + 2j|^p > 0$. Let

$$\boldsymbol{\alpha} := \frac{1}{\lambda} \cdot \left(\mathbf{1} + \frac{\boldsymbol{\alpha}'}{\|\boldsymbol{\alpha}'\|_\infty}\right) .$$

Then $\boldsymbol{\alpha}$ is non-negative, and $H_{k,p}(t^*) \cdot \boldsymbol{\alpha} = \mathbf{1} + \varepsilon \boldsymbol{e}_1$, where $\varepsilon = 1/(\lambda \cdot \|\boldsymbol{\alpha}'\|_\infty) > 0$, as needed. □

The main result of this section then follows by combining Lemma 3.1, Lemma 3.4, and Lemma 3.9.

Theorem 3.10. *For $k \in \mathbb{Z}^+$ and $p \in [1,\infty)$ if p satisfies either (1) $p \notin \mathbb{Z}$ or (2) $p \geq k$, there exists a (p,k)-isolating parallelepiped $V \in \mathbb{R}^{2^k \times k}$, $\boldsymbol{t}^* \in \mathbb{R}^{2^k}$. Moreover, if p is computable then there is an algorithm that on input k and p outputs such an isolating parallelepiped.*

3.4 A characterization of isolating parallelepipeds

By combining the new isolating parallelepiped construction for $p \notin \mathbb{Z}$ and $p \geq k$ and impossibility results in this paper (Theorem 3.10 and Corollary 6.4, respectively) with the isolating parallelepiped construction in [BGS17] for odd integer p, we obtain a complete characterization of the values of p and k for which there exist (p, k)-isolating parallelepipeds.

Theorem 3.11. *There exists a (p, k)-isolating parallelepiped for $k \in \mathbb{Z}^+$ and $p \in [1, \infty)$ if and only if p satisfies either (1) $p \notin 2\mathbb{Z}$ or (2) $p \geq k$. Moreover, there is an algorithm that on input $k \in \mathbb{Z}^+$ and any computable $p \in [1, \infty)$ with either (1) $p \notin 2\mathbb{Z}$ or (2) $p \geq k$, outputs $V \in \mathbb{R}^{2^k \times k}$ and $\boldsymbol{t}^* \in \mathbb{R}^{2^k}$ that define a (p, k)-isolating parallelepiped.*

Proof. By Proposition 4.4 and Corollary 4.7 in [BGS17], such parallelepipeds and the corresponding algorithm exist for odd integers p. Theorem 3.10 shows that such parallelepipeds exist for all $p \geq k$ and all $p \notin \mathbb{Z}$, with corresponding algorithms for computable p. Corollary 6.4 shows that these are the only cases in which isolating parallelepipeds exist. □

The reduction from (weighted Max-)k-SAT to CVP_p assuming the existence of computable (p, k)-isolating parallelepipeds given in [BGS17, Theorem 3.2] immediately implies the following. (We actually show a strictly stronger reduction in Section 5.)

Corollary 3.12. *For every $\varepsilon > 0$ and every computable $p \in [1, \infty) \setminus 2\mathbb{Z}$, there is no $2^{(1-\varepsilon)n}$-time algorithm for CVP_p assuming W-Max-SAT-SETH. In particular, there is no $2^{(1-\varepsilon)n}$-time algorithm for CVP_p assuming SETH.*

We also note that the "in particular" part of the above claim also holds for $p = \infty$ by [BGS17, Theorem 6.5], but that the reduction given in [BGS17, Theorem 3.2] only works when p is finite.

A natural question to ask is whether Corollary 3.12 can be extended to $p \in 2\mathbb{Z}$ using a reduction that does not use isolating parallelepipeds. In Section 6, we give an impossibility result precluding a much larger class of reductions, which we call "natural reductions."

4 Hardness of CVPP from on-off isolating parallelepipeds

In this section, we substantially improve the quantitative hardness results from [BGS17] for CVPP_p. [BGS17] showed $2^{\Omega(\sqrt{n})}$-hardness of CVPP_p for all $p \in [1, \infty)$ assuming non-uniform ETH, and did not show any additional hardness assuming non-uniform SETH. Here we show $2^{\Omega(n)}$-hardness of CVPP_p for all $p \neq 2$ (including even integers other than 2) assuming non-uniform ETH, and $2^{(1-\varepsilon)n}$-hardness of CVPP_p for all $p \notin 2\mathbb{Z}$ assuming non-uniform SETH. We also show both of these results for $p = \infty$. We do not show any improved hardness for the case where $p = 2$, which remains a tantalizing open question.

We show these results by defining a family of geometric gadgets called "(p, k)-on-off isolating parallepeipeds" that are defined by vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ and *two* targets $\boldsymbol{t}_{\text{on}}$ and $\boldsymbol{t}_{\text{off}}$, and then showing that such gadgets exist if and only if "normal" $(p, k+1)$-isolating parallepipeds exist. As the name suggests, (p, k)-on-off isolating parallelepipeds will allow us to "turn clauses on and off." More precisely, for a given n and k, we will output a single basis $B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ as preprocessing. Then, given a k-SAT instance Φ on n variables, we will output a target vector $\boldsymbol{t}$ that uses copies of $\boldsymbol{t}_{\text{on}}$ to "turn on" row blocks in B corresponding to all clauses in Φ, and copies of $\boldsymbol{t}_{\text{off}}$ to "turn off" row blocks in B corresponding to clauses not in Φ.

The high-level strategy of outputting a basis B that "represents all clauses possible in an n-variable k-SAT instance" as preprocessing, and then, given a k-SAT instance Φ on n variables, of "turning on and off clauses" according to whether they appear in Φ using the query target $\boldsymbol{t}$ is the same as was used in [BGS17, Lemma 6.1]. However, here we use a different framework for turning on and off clauses, and use it to output bases B of lower rank, leading to improved hardness results.

4.1 On-off isolating parallelepipeds

Definition 4.1 (On-off isolating parallelepiped)**.** *For $1 \leq p \leq \infty$ and $k \in \mathbb{Z}+$, we say that $V \in \mathbb{R}^{d^* \times k}$, $\boldsymbol{t}_{on} \in \mathbb{R}^{d^*}$, and $\boldsymbol{t}_{off} \in \mathbb{R}^{d^*}$ define a (p,k)-*on-off isolating parallelepiped *if:*

1. *For all $\boldsymbol{x} \in \{0,1\}^k \setminus \{\mathbf{0}\}$, $\|V\boldsymbol{x} - \boldsymbol{t}_{on}\|_p = 1$.*
2. *$\|V\mathbf{0} - \boldsymbol{t}_{on}\|_p = \|\boldsymbol{t}_{on}\|_p > 1$.*
3. *For all $\boldsymbol{x} \in \{0,1\}^k$, $\|V\boldsymbol{x} - \boldsymbol{t}_{off}\|_p = 1$.*[3]

We note that the first two conditions are the same as in the definition of "normal" isolating parallelepipeds (Definition 2.4) with $\boldsymbol{t}_{\text{on}}$ taking the role of $\boldsymbol{t}^*$. As in the case of isolating parallelepipeds, the $2^k - 1$ close vectors $V\boldsymbol{x}$ for $\boldsymbol{x} \in \{0,1\}^k \setminus \{\mathbf{0}\}$ to $\boldsymbol{t}_{\text{on}}$ correspond to the $2^k - 1$ possible satisfying assignments to the variables of a k-clause, and the more distant vector $\mathbf{0}$ corresponds to the single falsifying assignment to the variables of a k-clause. The new third condition asserts that all 2^k vectors $V\boldsymbol{x}$ for $\boldsymbol{x} \in \{0,1\}^k$ are equally close to $\boldsymbol{t}_{\text{off}}$, which says that the distance between $V\boldsymbol{x}$ and $\boldsymbol{t}_{\text{off}}$ will be the same regardless of whether the corresponding clause is satisfied or not. In other words, by using $\boldsymbol{t}_{\text{off}}$ in place of $\boldsymbol{t}_{\text{on}}$ (or $\boldsymbol{t}^*$), we will be able to "turn off" a clause so that its satisfiability is irrelevant.

The following proposition gives a construction of a (p,k)-on-off isolating parallelepiped from a $(p,k+1)$-isolating parallelepiped and vice-versa, therefore showing that one of these objects exists if and only if the other one does.

Proposition 4.2. *For every $p \in [1,\infty)$ and integer $k \geq 1$, there exists a computable (p,k)-on-off isolating parallelepiped if and only if there exists a computable $(p,k+1)$-isolating parallelepiped.*

Proof. Suppose that $V = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{k+1})$, $\boldsymbol{t}^*$ define a $(p,k+1)$-isolating parallelepiped. Set $V' := (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$, set $\boldsymbol{t}_{\text{on}} := \boldsymbol{t}^*$, and set $\boldsymbol{t}_{\text{off}} := \boldsymbol{t}^* - \boldsymbol{v}_{k+1}$. It is straightforward to check that $V', \boldsymbol{t}_{\text{on}}, \boldsymbol{t}_{\text{off}}$ define a (p,k)-on-off isolating parallelepiped.

Suppose that $V = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$, $\boldsymbol{t}_{\text{on}}$, $\boldsymbol{t}_{\text{off}}$ define a (p,k)-on-off isolating parallelepiped. Set $v'_i := v_i$ for $i = 1, \ldots, k$, set $v'_{k+1} := \boldsymbol{t}_{\text{on}} - \boldsymbol{t}_{\text{off}}$, and set $\boldsymbol{t}^* := \boldsymbol{t}_{\text{on}}$. It is straightforward to check that $V' := (v'_1, \ldots, v'_{k+1})$, $\boldsymbol{t}^*$ define a $(p,k+1)$-isolating parallelepiped. □

[3] It is natural to ask whether the given definition of an on-off isolating parallelepiped is sufficiently general. Indeed, one could define three different radii $r_{\text{good}} := \|V\boldsymbol{x} - \boldsymbol{t}_{\text{on}}\|_p$ for $\boldsymbol{x} \in \{0,1\}^k \setminus \{\mathbf{0}\}$, $r_{\text{bad}} := \|\boldsymbol{t}_{\text{on}}\|_p$, and $r_{\text{off}} := \|V\boldsymbol{x} - \boldsymbol{t}_{\text{off}}\|_p$ for $\boldsymbol{x} \in \{0,1\}^k$ corresponding to the three cases in the definition (with the requirement that $r_{\text{good}} < r_{\text{bad}}$). However, given $V, \boldsymbol{t}_{\text{on}}, \boldsymbol{t}_{\text{off}}$ satisfying these conditions for some $r_{\text{good}}, r_{\text{bad}}, r_{\text{off}}$, we can output another (p,k)-on-off isolating parallelepiped that achieves $r_{\text{off}} = r_{\text{good}} = 1$ simply by appending a coordinate of value $|r_{\text{good}}^p - r_{\text{off}}^p|^{1/p}$ to $\boldsymbol{t}_{\text{off}}$ if $r_{\text{good}} > r_{\text{off}}$ and to $\boldsymbol{t}_{\text{on}}$ if $r_{\text{off}} > r_{\text{good}}$, and then normalizing. So, the definition given is essentially without loss of generality.

4.2 Hardness of CVPP from on-off isolating parallelepipeds

The following theorem and corollary together say that, if there exists a (p,k)-on-off isolating parallelepiped for infinitely many k, then there is no $2^{(1-\varepsilon)n}$-time algorithm for CVPP_p assuming non-uniform SETH. They are analogous to Theorem 3.2 and Corollary 3.3 in [BGS17], but with "on-off isolating parallelepipeds" in place of "isolating parallelepipeds" and with "CVPP" in place of "CVP."[4] Theorem 4.3 also leads to $2^{\Omega(n)}$-hardness of CVP_p for all $p \neq 2$ assuming non-uniform ETH.

Theorem 4.3. *If there exists a computable (p,k)-on-off isolating parallelepiped defined by $V = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k) \in \mathbb{R}^{d^* \times k}$, $\boldsymbol{t}_{on} \in \mathbb{R}^{d^*}$, $\boldsymbol{t}_{off} \in \mathbb{R}^{d^*}$ for some $p \in [1,\infty)$ and $k \in \mathbb{Z}^+$, then there exist a pair of polynomial-time algorithms (P,Q) (in analogy to the definition of CVPP) that behave as follows.*

1. *On input $n \in \mathbb{Z}^+$, P outputs a basis $B \in \mathbb{R}^{d \times n}$ of a rank n lattice $\mathcal{L}$, where $d = 2^k \binom{n}{k} d^* + n$.*
2. *On input a Max-k-SAT instance with n variables, Q outputs a target vector $\boldsymbol{t} \in \mathbb{R}^d$ and a distance bound $r \geq 0$ such that $\mathrm{dist}_p(\boldsymbol{t}, \mathcal{L}) \leq r$ if and only if the input is a 'YES' instance.*

Proof. Let $M := 2^k \cdot \binom{n}{k} = O(n^k)$ be the total possible number of k-clauses on n variables, and let $C_1, \ldots, C_M$ denote those clauses. By assumption, there exists a (p,k)-isolating parallelepiped $V, \boldsymbol{t}_{\mathrm{on}}, \boldsymbol{t}_{\mathrm{off}}$ with $\|\boldsymbol{t}_{\mathrm{on}}\|_p = 1 + \varepsilon$ for some $\varepsilon > 0$.

The algorithm P constructs the basis $B \in \mathbb{R}^{d \times n}$ as

$$B := \begin{pmatrix} B_1 \\ \vdots \\ B_M \\ 2\alpha \cdot I_n \end{pmatrix},$$

for $\alpha := M^{1/p} \cdot (1+\varepsilon)$ and with blocks $B_i \in \mathbb{R}^{d^* \times n}$ defined by

$$(B_i)_j := \begin{cases} \boldsymbol{v}_s & \text{if } x_j \text{ is the } s\text{th literal of } C_i\ , \\ -\boldsymbol{v}_s & \text{if } \neg x_j \text{ is the } s\text{th literal of } C_i\ , \\ \boldsymbol{0} & \text{otherwise}\ , \end{cases}$$

for $1 \leq i \leq M$ and $1 \leq j \leq n$.

Given an instance (Φ, W) of Max-k-SAT with m clauses, the algorithm Q outputs $\boldsymbol{t} \in \mathbb{R}^d$ defined by

$$\boldsymbol{t} := \begin{pmatrix} \boldsymbol{t}_1 \\ \vdots \\ \boldsymbol{t}_M \\ \alpha \cdot \boldsymbol{1} \end{pmatrix},$$

[4] However, as a technical difference, the reduction below works as a reduction from MAX-k-SAT (or weighted MAX-k-SAT with polynomial integer weights), but not as a reduction from weighted MAX-k-SAT with arbitrary weights as in [BGS17, Theorem 3.2]. This is because the reduction in [BGS17, Theorem 3.2] requires scaling rows of both the basis matrix and target vector, and now we must output the basis matrix before we know the weights of the input weighted MAX-k-SAT instance.

where $\boldsymbol{t}_i := \boldsymbol{t}_{\text{on}} - \sum_{s \in N_i} \boldsymbol{v}_s$ if C_i is in Φ and $\boldsymbol{t}_i := \boldsymbol{t}_{\text{off}} - \sum_{s \in N_i} \boldsymbol{v}_s$ if C_i is not in Φ for $1 \le i \le M$, and

$$r := ((M - (m - W)) + (m - W) \cdot (1 + \varepsilon)^p + n \cdot \alpha^p)^{1/p} .$$

Clearly, both P and Q run in polynomial time. We next analyze for which $\boldsymbol{y} \in \mathbb{Z}^n$ it holds that $\|B\boldsymbol{y} - \boldsymbol{t}\|_p \le r$. Note that by the definition of α above, $\alpha^p = M \cdot (1+\varepsilon)^p \ge (M - (m-W)) + (m-W) \cdot (1+\varepsilon)^p$ for all m and W. Therefore, for $y \notin \{0,1\}^n$, $\|B\boldsymbol{y} - \boldsymbol{t}\|_p^p \ge \alpha^p \sum_{i=1}^n |2y_i - 1|^p \ge (n+2) \cdot \alpha^p > r^p$. So, we only need to analyze the case where $\boldsymbol{y} \in \{0,1\}^n$.

Consider an assignment $\boldsymbol{y} \in \{0,1\}^n$ to the variables of Φ. Then for $1 \le i \le M$ such that C_i is in Φ,

$$\begin{aligned}
\|B_i\boldsymbol{y} - \boldsymbol{t}_i\|_p &= \Big\| \sum_{s \in P_i} y_{\text{ind}(\ell_{i,s})} \cdot \boldsymbol{v}_s - \sum_{s \in N_i} y_{\text{ind}(\ell_{i,s})} \cdot \boldsymbol{v}_s - \Big(\boldsymbol{t}_{\text{on}} - \sum_{s \in N_i} \boldsymbol{v}_s\Big) \Big\|_p \\
&= \Big\| \sum_{s \in P_i} y_{\text{ind}(\ell_{i,s})} \cdot \boldsymbol{v}_s + \sum_{s \in N_i} \big(1 - y_{\text{ind}(\ell_{i,s})}\big) \cdot \boldsymbol{v}_s - \boldsymbol{t}_{\text{on}} \Big\|_p \\
&= \Big\| \sum_{s \in S_i(\boldsymbol{y})} \boldsymbol{v}_s - \boldsymbol{t}_{\text{on}} \Big\|_p .
\end{aligned}$$

By assumption, the last quantity is equal to 1 if $|S_i(\boldsymbol{y})| \ge 1$ and is equal to $1 + \varepsilon$ otherwise. A similar argument shows that for $1 \le i \le M$ such that C_i is not in Φ,

$$\|B_i\boldsymbol{y} - \boldsymbol{t}_i\|_p = \Big\| \sum_{s \in S_i(\boldsymbol{y})} \boldsymbol{v}_s - \boldsymbol{t}_{\text{off}} \Big\|_p = 1$$

regardless of $\boldsymbol{y}$.

Because $|S_i(\boldsymbol{y})| \ge 1$ if and only if C_i is satisfied, it follows that

$$\|B\boldsymbol{y} - \boldsymbol{t}\|_p^p = \Big(\sum_{i=1}^M \|B_i\boldsymbol{y} - \boldsymbol{t}_i\|_p^p \Big) + n \cdot \alpha^p = M - (m - m^+(\boldsymbol{y})) + (m - m^+(\boldsymbol{y})) \cdot (1 + \varepsilon)^p + n \cdot \alpha^p .$$

Therefore, $\|B\boldsymbol{y} - \boldsymbol{t}\|_p \le r$ if and only if $m^+(\boldsymbol{y}) \ge W$, and therefore there exists $\boldsymbol{y}$ such that $\|B\boldsymbol{y} - \boldsymbol{t}\|_p \le r$ if and only if (Φ, W) is a 'YES' instance of MAX-k-SAT, as needed. □

We get the following two corollaries about the hardness of CVPP_p assuming (non-uniform, Max-SAT versions of) SETH and ETH, respectively. Corollary 4.4 asserts that we get the same $2^{(1-\varepsilon)n}$ hardness of CVPP_p for $p \notin 2\mathbb{Z}$ that we get for CVP_p (assuming non-uniform SETH).

Corollary 4.4. *For every $p \in [1, \infty) \setminus 2\mathbb{Z}$ and $\varepsilon > 0$, there is no $2^{(1-\varepsilon)n}$-time algorithm for CVPP_p assuming non-uniform Max-SAT-SETH. In particular, there is no $2^{(1-\varepsilon)n}$-time algorithm for CVPP_p assuming non-uniform SETH.*

Proof. Combine Theorem 3.11, Proposition 4.2, and Theorem 4.3. □

Finally, Corollary 4.5 asserts that for *every* $p \ne 2$, CVPP_p takes $2^{\Omega(n)}$-time assuming ETH. We emphasize that, interestingly, this lower bound holds for even integers $p = 4, 6, \ldots$ greater than 2, therefore yielding a stronger hardness result for CVPP_p for *all* values of $p \ne 2$ than what is known for $p = 2$.

Corollary 4.5. *For every $p \ge 1$, $p \ne 2$, there is no $2^{o(n)}$-time algorithm for CVPP_p assuming non-uniform Max-SAT-ETH. In particular, there is no $2^{o(n)}$-time algorithm for CVPP_p assuming non-uniform ETH.*

4.3 SETH Hardness of CVPP_∞

Finally, we show that CVPP_∞ requires $2^{(1-\varepsilon)n}$-time assuming non-uniform SETH.

Theorem 4.6. *For every $k \in \mathbb{Z}^+$, there exists a pair of polynomial-time algorithms (P, Q) (in analogy to the definition of CVPP) that behave as follows.*

1. *On input $n \in \mathbb{Z}^+$, P outputs a basis $B \in \mathbb{R}^{d \times n}$ of a rank n lattice $\mathcal{L}$, where $d = 2^k \binom{n}{k} + n$.*
2. *On input a k-SAT instance with n variables, Q outputs a target vector $\boldsymbol{t} \in \mathbb{R}^d$ such that $\mathrm{dist}_\infty(\boldsymbol{t}, \mathcal{L}) \le k/2$ if and only if the input is a 'YES' instance.*

Proof. Let $M := 2^k \cdot \binom{n}{k} = O(n^k)$ be the total possible number of k-clauses on n variables, and let $C_1, \ldots, C_M$ denote those clauses.

The algorithm P constructs the basis $B \in \mathbb{R}^{d \times n}$ as

$$B := \begin{pmatrix} \boldsymbol{b}_1^T \\ \vdots \\ \boldsymbol{b}_M^T \\ k \cdot I_n \end{pmatrix} ,$$

and with rows $\boldsymbol{b}_i^T$ defined by

$$(B_i)_j := \begin{cases} 1 & \text{if } x_j \text{ is the } s\text{th literal of } C_i \ , \\ -1 & \text{if } \neg x_j \text{ is the } s\text{th literal of } C_i \ , \\ \mathbf{0} & \text{otherwise} \ , \end{cases}$$

for $1 \le i \le M$ and $1 \le j \le n$.

Given an instance Φ of k-SAT with m clauses, the algorithm Q outputs $\boldsymbol{t} \in \mathbb{R}^d$ defined by

$$\boldsymbol{t} := \begin{pmatrix} t_1 \\ \vdots \\ t_M \\ \frac{k}{2} \cdot \mathbf{1} \end{pmatrix} ,$$

where $t_i := (k+1)/2 - |N_i|$ if C_i is in Φ and $t_i := k/2 - |N_i|$ if C_i is not in Φ for $1 \le i \le M$, and where $r := k/2$.

Clearly, both P and Q run in polynomial time. We next analyze for which $\boldsymbol{y} \in \mathbb{Z}^n$ it holds that $\|B\boldsymbol{y} - \boldsymbol{t}\|_\infty \le r = k/2$. If $y \notin \{0,1\}^n$, $\|B\boldsymbol{y} - \boldsymbol{t}\|_\infty \ge \max_{i \in [n]} |y_i \cdot k - k/2| \ge 3k/2$. So, we only need to analyze the case where $\boldsymbol{y} \in \{0,1\}^n$.

Consider an assignment $\boldsymbol{y} \in \{0,1\}^n$ to the variables of Φ. Then for $1 \le i \le M$ such that C_i is in Φ,

$$\begin{aligned} \Big| \langle \boldsymbol{b}_i, \boldsymbol{y} \rangle - t_i \Big| &= \Big| \sum_{s \in P_i} y_{\mathrm{ind}(\ell_{i,s})} - \sum_{s \in N_i} y_{\mathrm{ind}(\ell_{i,s})} - ((k+1)/2 - |N_i|) \Big| \\ &= \Big| \sum_{s \in P_i} y_{\mathrm{ind}(\ell_{i,s})} - \sum_{s \in N_i} (1 - y_{\mathrm{ind}(\ell_{i,s})}) - (k+1)/2 \Big| \\ &= \Big| |S_i(\boldsymbol{y})| - (k+1)/2 \Big| . \end{aligned}$$

It follows that if $|S_i(\boldsymbol{y})| = 0$ then $|\langle \boldsymbol{b}_i, \boldsymbol{y}\rangle - t_i| = (k+1)/2$, and otherwise $|\langle \boldsymbol{b}_i, \boldsymbol{y}\rangle - t_i| \leq (k-1)/2$. Because $|S_i(\boldsymbol{y})| \geq 1$ if and only if clause C_i is satisfied, it follows that $|\langle \boldsymbol{b}_i, \boldsymbol{y}\rangle - t_i| \leq (k-1)/2$ if and only if clause C_i is satisfied.

A similar argument shows that for $1 \leq i \leq M$ such that C_i is not in Φ,

$$|\langle \boldsymbol{b}_i, \boldsymbol{y}\rangle - t_i| = ||S_i(\boldsymbol{y})| - k/2|$$

regardless of $\boldsymbol{y}$.

Therefore for $\boldsymbol{y} \in \{0,1\}^n$, $\max_{1\leq i\leq M} |\langle \boldsymbol{b}_i, \boldsymbol{y}\rangle - t_i|$ is less than or equal to $k/2$ if every clause in Φ is satisfied, and is greater than $(k+1)/2$ if there exists a clause in Φ that is not satisfied. It follows that

$$\|B\boldsymbol{y} - \boldsymbol{t}\|_\infty = \max\{|\langle \boldsymbol{b}_1, \boldsymbol{y}\rangle - t_1|, \ldots, |\langle \boldsymbol{b}_m, \boldsymbol{y}\rangle - t_m|, k/2\} = k/2 = r$$

if $\boldsymbol{y}$ satisfies Φ, and $\|B\boldsymbol{y} - \boldsymbol{t}\|_\infty \geq (k+1)/2 > r$ if not. Therefore, there exists $\boldsymbol{y} \in \{0,1\}^n$ that satisfies Φ if and only if there exists $\boldsymbol{y} \in \{0,1\}^n$ that satisfies $\|B\boldsymbol{y} - \boldsymbol{t}\|_\infty$, as needed. □

We note that the preceding reduction actually gives mild hardness of approximation for CVPP_∞ (depending on k), which is similar to the case for CVP_∞ [BGS17, Theorem 6.5].

Corollary 4.7. *For every $\varepsilon > 0$, there exists $k = k(\varepsilon) \in \mathbb{Z}^+$ such that there is no $2^{(1-\varepsilon)n}$-time algorithm that approximates CVPP_∞ to within a factor less than $1 + 1/k$ assuming non-uniform SETH. In particular, there is no $2^{(1-\varepsilon)n}$-time algorithm for $CVPP_\infty$ assuming non-uniform SETH.*

5 Gap-SETH hardness of CVP

In this section, we show that for all $p \in [1,\infty) \setminus 2\mathbb{Z}$ and every $\varepsilon > 0$ there exists $\gamma = \gamma(p,\varepsilon) > 1$ such that there is no $2^{(1-\varepsilon)n}$-time algorithm for γ-approximate CVP_p (and CVPP_p) assuming Gap-SETH (Definition 2.7).

The main reduction in [BGS17] reduces k-SAT instances with n variables to CVP instances of rank n with basis matrix B where closest lattice vectors are guaranteed to be 0-1 combinations of basis vectors. These 0-1 combinations naturally correspond to boolean assignments to the variables of the k-SAT formula, which is essential to the analysis of the reduction. That reduction in [BGS17] enforces the condition that 0-1 combinations of vectors in the basis B are closest to the target $\boldsymbol{t}$ by appending a scaled identity matrix $2\alpha \cdot I_n$ for some large $\alpha > 0$ to the bottom of another matrix B', and appending the "all α" vector $\alpha \cdot \mathbf{1}$ to the bottom of another vector $\boldsymbol{t}'$ (the hardness reduction for CVPP_p in Theorem 4.3 also works this way).

Because of the scaled identity matrix appended to the bottom of B', the ratio between the distance of 0-1 combinations of basis vectors corresponding to satisfying and unsatisfying assignments approaches 1 as n approaches infinity, even when k is fixed. This precludes the reduction working as a reduction from Gap-k-SAT to γ-approximate CVP for γ independent of n. However, in this section, we show that by appending $2\alpha \cdot I_k$, $\alpha \cdot \mathbf{1}$ to the respective components $V = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$, $\boldsymbol{t}^*$ of an isolating parallelepiped instead of appending $2\alpha \cdot I_n$, $\alpha \cdot \mathbf{1}$ to the bottom of B', $\boldsymbol{t}'$ lets us circumvent this issue and therefore prove stronger (conditional) hardness of approximation results for CVP_p.

5.1 Isolating lattices

The following definition strengthens the notion of an "isolating parallelepiped" to an "isolating lattice," which is defined by a basis matrix $V \in \mathbb{R}^{d^* \times k}$ and a target $\boldsymbol{t}^* \in \mathbb{R}^{d^*}$. Like for an isolating parallelepiped, we require that $\|\boldsymbol{t}^*\| > \|V\boldsymbol{x} - \boldsymbol{t}^*\| = 1$ for all non-zero $\boldsymbol{x} \in \{0,1\}^k$, but for isolating lattices we also require that $\|V\boldsymbol{x} - \boldsymbol{t}^*\| > \|\boldsymbol{t}^*\|$ for $\boldsymbol{x} \notin \{0,1\}^k$. We also require V to have linearly independent columns (so that it forms a basis) rather than allowing it to be arbitrary.

Definition 5.1. *For any $1 \le p \le \infty$ and integer $k \ge 1$, we say that $V \in \mathbb{R}^{d^* \times k}$ with full column rank and $\boldsymbol{t}^* \in \mathbb{R}^{d^*}$ define a* (p,k)-isolating lattice *if there exists $\varepsilon > 0$ such that:*

1. $\|V\boldsymbol{x} - \boldsymbol{t}^*\|_p = 1$ *for all* $\boldsymbol{x} \in \{0,1\}^k \setminus \{\boldsymbol{0}\}$,
2. $\|\boldsymbol{t}^*\|_p = 1 + \varepsilon$,
3. $\|V\boldsymbol{x} - \boldsymbol{t}^*\|_p > 1 + \varepsilon$ *for all* $\boldsymbol{x} \in \mathbb{Z}^k \setminus \{0,1\}^k$.

The following proposition shows how to construct a (p,k)-isolating lattice from any (p,k)-isolating parallelepiped $V, \boldsymbol{t}^*$. Again, the idea is simply to append a scaled identity matrix to the bottom of V and a vector whose entries are all the same to the bottom of $\boldsymbol{t}^*$.

Proposition 5.2. *For every $p \in [1,\infty)$ and $k \in \mathbb{Z}^+$, there exists a computable (p,k)-isolating lattice if and only if there exists a computable (p,k)-isolating parallelepiped.*

In order to show this, we will use the following claim.

Claim 5.3. *Suppose that $V = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$, $\boldsymbol{t}^*$ define a (p,k)-isolating parallelepiped for $1 \le p \le \infty$ and $k \ge 2$. Then $\|\boldsymbol{t}^*\|_p \le 3$.*

Proof. By the triangle inequality and the definition of an isolating parallelepiped, $\|\boldsymbol{v}_1\|_p \le \|\boldsymbol{v}_2 - \boldsymbol{t}^*\|_p + \|\boldsymbol{v}_1 + \boldsymbol{v}_2 - \boldsymbol{t}^*\|_p = 2$ and so $\|\boldsymbol{t}^*\|_p \le \|\boldsymbol{v}_1\|_p + \|\boldsymbol{v}_1 - \boldsymbol{t}^*\|_p \le 3$. □

We note that the above claim is tight for the simple, degenerate 1-dimensional $(p,2)$-isolating paralellepiped where $\boldsymbol{v}_1 := \boldsymbol{v}_2 := 2$ and $\boldsymbol{t}^* := 3$ (such isolating parallelepipeds were used in [BGS17]).

Proof of Proposition 5.2. Every (p,k)-isolating lattice is already a (p,k)-isolating parallelepiped by definition. On the other hand, suppose that V, $\boldsymbol{t}^*$ define a (p,k)-isolating parallelepiped. Then set

$$V' := \begin{pmatrix} V \\ 6 \cdot I_k \end{pmatrix}, \qquad \boldsymbol{t}' := \begin{pmatrix} \boldsymbol{t}^* \\ 3 \cdot \boldsymbol{1} \end{pmatrix}.$$

It is straightforward to check that V' has full column rank, that $\|V'\boldsymbol{x} - \boldsymbol{t}'\|_p^p = k \cdot 3^p + 1$ for $\boldsymbol{x} \in \{0,1\}^k \setminus \{\boldsymbol{0}\}$, that $k \cdot 3^p + 1 < \|\boldsymbol{t}'\|_p^p \le (k+1) \cdot 3^p$ (by Claim 5.3), and that $\|V'\boldsymbol{x} - \boldsymbol{t}'\|_p^p \ge (k+1) \cdot 3^p + 9^p > (k+1) \cdot 3^p \ge \|\boldsymbol{t}'\|_p^p$ for $\boldsymbol{x} \in \mathbb{Z}^k \setminus \{0,1\}^k$. Therefore, we can normalize $V', \boldsymbol{t}'$ to obtain a (p,k)-isolating lattice. □

5.2 Gap-SETH hardness of CVP from isolating lattices

Theorem 5.4. *If there exists a computable (p,k)-isolating lattice for some $p \in [1,\infty)$ and $k \in \mathbb{Z}^+$, then there exists a polynomial time reduction from any $(1-\delta,1)$-Gap-k-SAT instance with n variables and $\delta \in (0,1)$ to a γ-CVP$_p$ instance of rank n with $\gamma = \gamma(p,k,\delta) > 1$.*

Proof. Let Φ be a $(1-\delta,1)$-Gap-k-SAT instance with n variables and m clauses $C_1,\ldots,C_m$. Suppose that $V \in \mathbb{R}^{d^* \times k}$, $\boldsymbol{t}^* \in \mathbb{R}^{d^*}$ define a (p,k)-isolating lattice, with $\|\boldsymbol{t}^*\| = 1+\varepsilon$ for some $\varepsilon > 0$ as in Definition 5.1.

We define the output γ-CVP$_p$ instance $(B,\boldsymbol{t},r)$ as follows. We set

$$B := \begin{pmatrix} B_1 \\ \vdots \\ B_m \end{pmatrix}, \qquad \boldsymbol{t} := \begin{pmatrix} \boldsymbol{t}_1 \\ \vdots \\ \boldsymbol{t}_m \end{pmatrix},$$

with blocks $B_i \in \mathbb{R}^{d^* \times n}$ defined by

$$(B_i)_j := \begin{cases} \boldsymbol{v}_s & \text{if } x_j \text{ is the } s\text{th literal of } C_i, \\ -\boldsymbol{v}_s & \text{if } \neg x_j \text{ is the } s\text{th literal of } C_i, \\ \boldsymbol{0} & \text{otherwise} \end{cases}$$

for $1 \le i \le m$ and $1 \le j \le n$, and $\boldsymbol{t}_i := \boldsymbol{t}^* - \sum_{s \in N_i} \boldsymbol{v}_s$. We set $r := m^{1/p}$. Clearly, the reduction runs in polynomial time. The fact that B is full-rank (and hence a lattice basis) follows from the fact that V is full-rank, assuming without loss of generality that all n variables appear in Φ.

Given $\boldsymbol{y} \in \mathbb{Z}^n$, let $\chi(\boldsymbol{y}) \in \{0,1\}^n$ denote the vector whose ith coordinate is set to 1 if $y_i \ge 1$ and is set to 0 otherwise. Fix such a $\boldsymbol{y} \in \mathbb{Z}^n$. Then

$$\begin{aligned} \|B_i\boldsymbol{y} - \boldsymbol{t}_i\|_p &= \Big\| \sum_{s \in P_i} y_{\mathrm{ind}(\ell_{i,s})} \cdot \boldsymbol{v}_s - \sum_{s \in N_i} y_{\mathrm{ind}(\ell_{i,s})} \cdot \boldsymbol{v}_s - \Big(\boldsymbol{t}^* - \sum_{s \in N_i} \boldsymbol{v}_s\Big) \Big\|_p \\ &= \Big\| \sum_{s \in P_i} y_{\mathrm{ind}(\ell_{i,s})} \cdot \boldsymbol{v}_s + \sum_{s \in N_i} \big(1 - y_{\mathrm{ind}(\ell_{i,s})}\big) \cdot \boldsymbol{v}_s - \boldsymbol{t}^* \Big\|_p \\ &\ge \Big\| \sum_{s \in P_i} \chi(\boldsymbol{y})_{\mathrm{ind}(\ell_{i,s})} \cdot \boldsymbol{v}_s + \sum_{s \in N_i} \big(1 - \chi(\boldsymbol{y})_{\mathrm{ind}(\ell_{i,s})}\big) \cdot \boldsymbol{v}_s - \boldsymbol{t}^* \Big\|_p \\ &= \Big\| \sum_{s \in S_i(\chi(\boldsymbol{y}))} \boldsymbol{v}_s - \boldsymbol{t}^* \Big\|_p . \end{aligned} \tag{4}$$

We consider two cases: (1) the case where $\boldsymbol{y} \in \{0,1\}^n$, and (2) the case where $\boldsymbol{y} \notin \{0,1\}^n$.

In case (1), the inequality in Equation (4) is an equality, and we have that $\|B_i\boldsymbol{y} - \boldsymbol{t}_i\|_p = \Big\| \sum_{s \in S_i(\chi(\boldsymbol{y}))} \boldsymbol{v}_s - \boldsymbol{t}^* \Big\|_p = \| \sum_{s \in S_i(\boldsymbol{y})} \boldsymbol{v}_s - \boldsymbol{t}^* \|_p$, which is equal to 1 if $\boldsymbol{y}$ satisfies C_i (i.e. if $|S_i(\boldsymbol{y})| \ge 1$), and is equal to $1+\varepsilon$ otherwise. Therefore, for $\boldsymbol{y} \in \{0,1\}^n$,

$$\|B\boldsymbol{y} - \boldsymbol{t}\|_p^p = \sum_{i=1}^m \|B_i\boldsymbol{y} - \boldsymbol{t}_i\|_p^p = m^+(\boldsymbol{y}) + (m - m^+(\boldsymbol{y})) \cdot (1+\varepsilon)^p .$$

In case (2), the inequality in Equation (4) is strict by the definition of an isolating lattice, and so we have that $\|B_i\boldsymbol{y} - \boldsymbol{t}_i\|_p > \|B_i \cdot \chi(\boldsymbol{y}) - \boldsymbol{t}_i\|_p = \Big\| \sum_{s \in S_i(\chi(\boldsymbol{y}))} \boldsymbol{v}_s - \boldsymbol{t}^* \Big\|_p$ for all $1 \le i \le m$.

It follows that if $\mathrm{val}(\Phi) = 1$, then there exists $\boldsymbol{y} \in \{0,1\}^n$ such that $\|B\boldsymbol{y} - \boldsymbol{t}\|_p^p \le m = r^p$, and if $\mathrm{val}(\Phi) < 1 - \delta$, then for every $\boldsymbol{y} \in \mathbb{Z}^n$, $\|B\boldsymbol{y} - \boldsymbol{t}\|_p^p \ge \|B \cdot \chi(\boldsymbol{y}) - \boldsymbol{t}\|_p^p > \delta m \cdot (1+\varepsilon)^p + (1-\delta)m = (\delta(1+\varepsilon)^p + (1-\delta)) \cdot r^p$. Therefore, the output is an instance of γ-CVP with

$$\gamma = \gamma(p, k, \delta) := (\delta \cdot (1+\varepsilon)^p + (1-\delta))^{1/p} > 1 \ , \tag{5}$$

which is a 'YES' instance if Φ is a 'YES' instance and a 'NO' instance if Φ is a 'NO' instance, as needed. □

Corollary 5.5. *For all $p \in [1,\infty) \setminus 2\mathbb{Z}$ and every $\varepsilon > 0$ there exists $\gamma = \gamma(p,\varepsilon) > 1$ such that there is no $2^{(1-\varepsilon)n}$-time algorithm for γ-CVP_p assuming Gap-SETH.*

Proof. Combine Theorem 3.11, Proposition 5.2, and Theorem 5.4. □

Finally, we note that we can extend Theorem 5.4 to give a reduction from *arbitrary* Gap Constraint Satisfaction Problems (Gap-CSPs) to CVP_p. This leads to a promising approach for proving even stronger quantitative hardness of approximation results for CVP_p via a reduction from Gap-k-CSPs other than Gap-k-SAT. Namely, it is known that general k-CSPs are hard to approximate to within much smaller approximation factors than k-SAT [Cha16, AM09, MM17], and if one were to hypothesize some quantitative hardness of approximation for them then we could conclude corresponding quantitative hardness of approximation results about CVP_p as follows.

For a constraint $C : \{0,1\}^k \to \{0,1\}$, we define a (p,k)-C-isolating lattice $V, \boldsymbol{t}^*$ as a generalization of an isolating lattice with conditions (1) and (2) in Definition 5.1 replaced by (1) $\|V\boldsymbol{x} - \boldsymbol{t}^*\|_p = 1$ when $\boldsymbol{x} \in C^{-1}(1)$, and (2) $\|V\boldsymbol{x} - \boldsymbol{t}^*\|_p = 1 + \varepsilon$ for some $\varepsilon > 0$ when $\boldsymbol{x} \in C^{-1}(0)$, respectively. We can then use these C-isolating lattices in place of "normal" isolating lattices in Theorem 5.4.

Of course, for this reduction to work, we need to show how to construct C-isolating lattices. We can easily do this (for $p \notin \mathbb{Z}$ and $p \le k$) by running the argument in Lemma 3.9 with the complemented truth table of an arbitrary constraint C in place of $\boldsymbol{e}_1$ to obtain a "C-isolating parallelepiped," and then using Proposition 5.2 to convert it into a C-isolating lattice. However, it is not clear how to lower bound the ε that we obtain from this reduction explicitly. (Showing that this ε, which depends on C, is large enough is necessary for proving quantitative hardness results with explicit approximation factors.)

6 Limitations

6.1 Impossibility of (p,k)-isolating parallelepipeds for even integer $p < k$

In [BGS17], we proved that there do not exist $(2,3)$-isolating parallelepipeds, and noted that there are no $(p, p+1)$-isolating parallelepipeds for $p \in 2\mathbb{Z}$. Here, we give a simple geometric proof of the non-existence of $(2,3)$-isolating parallelepipeds, and we also prove that there are no $(p, p+1)$-isolating parallelepipeds for $p \in 2\mathbb{Z}$. This finishes the complete characterization of values of p and k such that (p,k)-isolating parallelepipeds exist, as presented in Theorem 3.11.

Lemma 6.1. *Suppose that $V = (\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3) \in \mathbb{R}^{d \times 3}$, $\boldsymbol{t} \in \mathbb{R}^d$, and $\|V\boldsymbol{x} - \boldsymbol{t}\| = 1$ for all $\boldsymbol{x} \in \{0,1\}^3 \setminus \{\boldsymbol{0}\}$. Then $\|\boldsymbol{t}\| = \|V\boldsymbol{x} - \boldsymbol{t}\|$ for $\boldsymbol{x} \in \{0,1\}^3 \setminus \{\boldsymbol{0}\}$, and hence $V, \boldsymbol{t}$ do not form an isolating parallelepiped.*

Proof. For $\boldsymbol{p} = \boldsymbol{t} - \boldsymbol{v}_3$, by assumption we have that

$$\|\boldsymbol{p}\| = \|\boldsymbol{v}_1 - \boldsymbol{p}\| = \|\boldsymbol{v}_2 - \boldsymbol{p}\| = \|\boldsymbol{v}_1 + \boldsymbol{v}_2 - \boldsymbol{p}\| = r \ .$$

Let us consider a plane P passing through the points $\mathbf{0}, \boldsymbol{v}_1$ and $\boldsymbol{v}_2$, and let $\boldsymbol{p}^*$ be the projection of $\boldsymbol{p}$ onto P. Consider the parallelogram D formed by the points $\mathbf{0}, \boldsymbol{v}_1, \boldsymbol{v}_2$ and $\boldsymbol{v}_1 + \boldsymbol{v}_2$. These points lie on a circle around the point $\boldsymbol{p}^*$. Therefore, D is a cyclic parallelogram, i.e., a rectangle.

Let $\boldsymbol{t}^*$ be the projection of $\boldsymbol{t}$ onto P. Let $\|\boldsymbol{v}_1 - \boldsymbol{t}^*\| = \|\boldsymbol{v}_2 - \boldsymbol{t}^*\| = \|\boldsymbol{v}_1 + \boldsymbol{v}_2 - \boldsymbol{t}^*\| = r'$. Since the three points of the rectangle formed by the points $\mathbf{0}, \boldsymbol{v}_1, \boldsymbol{v}_2$ and $\boldsymbol{v}_1 + \boldsymbol{v}_2$ lie on the circle of radius $r'/2$ around the point $\boldsymbol{t}^*$, the fourth point of this rectangle also lies on this circle. Thus, $\|\boldsymbol{t}\| = \|\boldsymbol{v}_1 - \boldsymbol{t}\|$. □

Corollary 6.2. *There do not exist $(2, k)$-isolating parallelepipeds for $k \geq 3$.*

Lemma 6.3. *For every $p \in 2\mathbb{Z}$, integers d and $k > p$, and vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k, \boldsymbol{t} \in \mathbb{R}^d$, we have*

$$\sum_{S \subseteq [k]} (-1)^{|S|} \left\| \boldsymbol{t} - \sum_{i \in S} \boldsymbol{v}_i \right\|_p^p = 0 \ .$$

Proof. We will use the Multinomial theorem which states that

$$(x_1 + \ldots + x_m)^n = \sum_{a_1 + \ldots + a_m = n} \binom{n}{a_1, \ldots, a_m} \prod_{t=1}^{m} x_t^{a_t} \ ,$$

where

$$\binom{n}{a_1, \ldots, a_m} = \frac{n!}{a_1! \cdots a_m!} \ .$$

Let $\boldsymbol{t} = (t_1, \ldots, t_d)$ and for an $i \in [k], \boldsymbol{v}_i = (v_{i,1}, \ldots, v_{i,d})$. For a set $S \in [k]$, and an integer $1 \leq i \leq |S|$, let S_i be the ith element of the set S. Then we have that for $p \in 2\mathbb{Z}$,

$$\begin{aligned}
\sum_{S \subseteq [k]} (-1)^{|S|} \left\| \boldsymbol{t} - \sum_{i \in S} \boldsymbol{v}_i \right\|_p^p &= \sum_{S \subseteq [k]} (-1)^{|S|} \sum_{j=1}^{d} (t_j - \sum_{i=1}^{|S|} v_{S_i, j})^p \\
&= \sum_{S \subseteq [k]} (-1)^{|S|} \sum_{j=1}^{d} \sum_{a_0 + \ldots + a_{|S|} = p} \binom{p}{a_0, \ldots, a_{|S|}} t_j^{a_0} \prod_{i=1}^{|S|} v_{S_i, j}^{a_i} \\
&= \sum_{j=1}^{d} \sum_{a_0 + \ldots + a_k = p} \binom{p}{a_0, \ldots, a_k} t_j^{a_0} \prod_{i=1}^{k} v_{i,j}^{a_i} \cdot \sum_{S \supseteq \{i \colon a_i \neq 0\}} (-1)^{|S|} \\
&= \sum_{j=1}^{d} \sum_{a_0 + \ldots + a_k = p} \binom{p}{a_0, \ldots, a_k} t_j^{a_0} \prod_{i=1}^{k} v_{i,j}^{a_i} \cdot (1-1)^{k - |\{i \colon a_i \neq 0\}|} \\
&= 0 \ ,
\end{aligned}$$

where the last equality follows from $|\{i \colon a_i \neq 0\}| \leq p < k$. □

Corollary 6.4. *Let $p \in 2\mathbb{Z}$. There do not exist (p, k)-isolating parallelepipeds for $k > p$.*

Proof. Suppose towards a contradiction that $V = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k) \in \mathbb{R}^{d \times k}$ and $\boldsymbol{t} \in \mathbb{R}^d$ form an isolating parallelepiped. Then for all $\boldsymbol{x} \in \{0,1\}^k \setminus \{\mathbf{0}\}$, $\|V\boldsymbol{x} - \boldsymbol{t}\| = 1$. By Lemma 6.1,

$$\|\boldsymbol{t}\|_p^p = \sum_{\emptyset \neq S \subseteq [k]} (-1)^{|S|+1} \|t - \sum_{i \in S} v_i\|_p^p = \sum_{\emptyset \neq S \subseteq [k]} (-1)^{|S|+1} = 1 \ .$$ □

6.2 Impossibility of natural reductions for $p = 2$

For a lattice $\mathcal{L} \subset \mathbb{R}^d$ with basis $\mathbf{B} \in \mathbb{R}^{d \times n}$ and target vector $\boldsymbol{t} \in \mathbb{R}^d$, let

$$\mathrm{CVP}(\boldsymbol{t}, \mathbf{B}) := \{\boldsymbol{z} \in \mathbb{Z}^n \ : \ \|\mathbf{B}\boldsymbol{z} - \boldsymbol{t}\|_2 = \mathrm{dist}_2(\boldsymbol{t}, \mathcal{L})\}$$

be the set of the coordinates of closest lattice vectors to $\boldsymbol{t}$.

Definition 6.5. *A* natural reduction *from 3-SAT to* CVP_2 *is a (not necessarily efficient) reduction from 3-SAT instances on n variables to* CVP_2 *instances $\mathbf{B} \in \mathbb{R}^{d \times n'}, \boldsymbol{t} \in \mathbb{R}^d$ such that there exists a (not necessarily efficient) function $f : \{0,1\}^n \to \mathbb{Z}^{n'}$ with the following property. If the input 3-SAT instance is satisfiable, then for every $\boldsymbol{x} \in \{0,1\}^n$, $\boldsymbol{x}$ is a satisfying assignment if and only if $f(\boldsymbol{x}) \in \mathrm{CVP}(\boldsymbol{t}, \mathbf{B})$.*

The following theorem shows that no natural reduction can rule out a $2^{3n/4}$-time algorithm for CVP_2 under SETH.

Theorem 6.6. *Every natural reduction from 3-SAT on n variables to* CVP_2 *on rank n' lattices must have $n' > 4(n-2)/3$.*

To prove Theorem 6.6, we study the structure of $A := f(\{0,1\}^n)$ modulo two. In particular, we will show that A cannot contain any affine 3-cube modulo two. The next lemma is a version of Szemerédi's cube lemma for the boolean cube, which shows that any such set must be small (relative to n'). To the authors' knowledge, our proof is novel and significantly simpler than that of prior work (e.g., [CS16, Lemma 3.1]). We also obtain a tighter bound.

Lemma 6.7. *Let $d \geq 1$ be an integer. Every set $S \subseteq \mathbb{F}_2^n$ of size $|S| \geq 2^{n(1-2^{-(d-1)})+2}$ contains an affine subspace of dimension d.*

Proof. We prove the result by induction on d. For $d = 1$, we have $|S| \geq 4$, and so the statement is trivially true since any set with 2 elements contains an affine subspace of dimension 1.

Now we assume the result is true for $d = k$, and show that it is true for $d = k + 1 \geq 2$. Let $S := \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, where $N = |S| \geq 2^{n(1-2^{-k})+2}$. Consider all $\binom{N}{2}$ distinct pairs of elements in S. By the pigeon-hole principle, at least

$$M = \frac{N(N-1)}{2 \cdot 2^n} \geq \frac{N^2}{4 \cdot 2^n} = 2^{n(1-2^{-(k-1)})+2}$$

distinct pairs have the same sum, say $\boldsymbol{z}_0 \in \mathbb{F}_2^n$. Without loss of generality, let these pairs be $(\boldsymbol{x}_1, \boldsymbol{x}_1 + \boldsymbol{z}_0), (\boldsymbol{x}_2, \boldsymbol{x}_2 + \boldsymbol{z}_0), \ldots, (\boldsymbol{x}_M, \boldsymbol{x}_M + \boldsymbol{z}_0)$.

By the induction hypothesis, there exist $\boldsymbol{z}^*$, and linearly independent vectors $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_k$ such that the set $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$ contains every element of the form $\boldsymbol{z}^* + \sum_{i=1}^k \sigma_i \boldsymbol{z}_i$ where $\sigma_i \in \{0,1\}$ for $1 \leq i \leq k$.

This implies that S contains every element of the form $\boldsymbol{z}^* + \sum_{i=0}^k \sigma_i \boldsymbol{z}_i$ where $\sigma_i \in \{0,1\}$ for $0 \leq i \leq k$. To complete the proof, we need to show that $\boldsymbol{z}_0$ is not in the span of $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_k$. But this is immediate from the fact that each of the M pairs above contains distinct elements. □

This next lemma shows that the coordinates of closest vectors have some additional structure modulo two. In particular, if $\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \boldsymbol{z}_4 \in \mathrm{CVP}(\boldsymbol{t}, \mathbf{B})$ form a square modulo two (i.e., a two-dimensional affine subspace), then either they form a parallelogram over the reals or there must be some specific set of four other vectors $\boldsymbol{z}'_1, \boldsymbol{z}'_2, \boldsymbol{z}'_3, \boldsymbol{z}'_4 \in \mathrm{CVP}(\boldsymbol{t}, \mathbf{B})$. We will then use this to argue that $A := f(\{0,1\}^n)$ cannot contain any affine 3-cubes modulo two.

Lemma 6.8. *For any lattice $\mathcal{L} \subset \mathbb{R}^d$ with rank $n \geq 2$ and basis $\mathbf{B} \in \mathbb{R}^{d \times n}$ and any target $\boldsymbol{t} \in \mathbb{R}^d$, suppose that $\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \boldsymbol{z}_4 := \boldsymbol{z}_1 + \boldsymbol{z}_2 + \boldsymbol{z}_3 - 2\boldsymbol{v} \in \mathrm{CVP}(\boldsymbol{t}, \mathbf{B})$ are coordinates of distinct closest lattice vectors with $\boldsymbol{v} \in \mathbb{Z}^n$. Then $\boldsymbol{z}'_1, \boldsymbol{z}'_2, \boldsymbol{z}'_3, \boldsymbol{z}'_4 \in \mathrm{CVP}(\boldsymbol{t}, \mathbf{B})$ where*

$$\boldsymbol{z}'_1 := \boldsymbol{z}_2 + \boldsymbol{z}_3 - \boldsymbol{v}, \quad \boldsymbol{z}'_2 := \boldsymbol{z}_1 + \boldsymbol{z}_3 - \boldsymbol{v}, \quad \boldsymbol{z}'_3 := \boldsymbol{z}_1 + \boldsymbol{z}_2 - \boldsymbol{v}, \quad \boldsymbol{z}'_4 := \boldsymbol{v} \ .$$

In particular, $C := \{\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \boldsymbol{z}_4\} \cup \{\boldsymbol{z}'_1, \boldsymbol{z}'_2, \boldsymbol{z}'_3, \boldsymbol{z}'_4\}$ has size either four or eight, and $|C| = 4$ if and only if $C = \{\boldsymbol{y}_0, \boldsymbol{y}_0 + \boldsymbol{y}_1, \boldsymbol{y}_0 + \boldsymbol{y}_2, \boldsymbol{y}_0 + \boldsymbol{y}_1 + \boldsymbol{y}_2\}$ for some $\boldsymbol{y}_i \in \mathbb{Z}^n$, i.e., C is a parallelogram.

Proof. By shifting $\boldsymbol{t}$ appropriately, we may assume without loss of generality that $\boldsymbol{z}_3 = \mathbf{0}$. Let $\boldsymbol{x} := \mathbf{B}\boldsymbol{z}_1$, $\boldsymbol{y} := \mathbf{B}\boldsymbol{z}_2$, and $\boldsymbol{w} := \mathbf{B}\boldsymbol{v}$. Since $\mathbf{0}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{x} + \boldsymbol{y} - 2\boldsymbol{w}$ are all the same distance from $\boldsymbol{t}$, we have

$$\|\boldsymbol{x} - \boldsymbol{t}\|_2^2 = \|\boldsymbol{t}\|_2^2 \ , \qquad \|\boldsymbol{y} - \boldsymbol{t}\|_2^2 = \|\boldsymbol{t}\|_2^2 \ , \qquad \|\boldsymbol{x} + \boldsymbol{y} - 2\boldsymbol{w} - \boldsymbol{t}\|_2^2 = \|\boldsymbol{t}\|_2^2 \ .$$

Recalling the identity $\|\boldsymbol{u}_1 - \boldsymbol{u}_2\|_2^2 = \|\boldsymbol{u}_1\|_2^2 + \|\boldsymbol{u}_2\|_2^2 - 2\langle \boldsymbol{u}_1, \boldsymbol{u}_2 \rangle$, we have

$$0 = \|\boldsymbol{x}\|_2^2 - 2\langle \boldsymbol{x}, \boldsymbol{t} \rangle = \|\boldsymbol{y}\|_2^2 - 2\langle \boldsymbol{y}, \boldsymbol{t} \rangle = \|\boldsymbol{x} + \boldsymbol{y} - 2\boldsymbol{w}\|_2^2 - 2\langle \boldsymbol{x}, \boldsymbol{t} \rangle - 2\langle \boldsymbol{y}, \boldsymbol{t} \rangle + 4\langle \boldsymbol{w}, \boldsymbol{t} \rangle \ . \tag{6}$$

Furthermore, since $\mathbf{0} \in \mathrm{CVP}(\boldsymbol{t}, \mathcal{L})$, and since $\boldsymbol{w}, \boldsymbol{x} - \boldsymbol{w}, \boldsymbol{y} - \boldsymbol{w}, \boldsymbol{x} + \boldsymbol{y} - \boldsymbol{w}$ are lattice vectors, we must have

$$\|\boldsymbol{w} - \boldsymbol{t}\|_2^2 \geq \|\boldsymbol{t}\|_2^2 \ , \quad \|\boldsymbol{x} - \boldsymbol{w} - \boldsymbol{t}\|_2^2 \geq \|\boldsymbol{t}\|_2^2 \ , \quad \|\boldsymbol{y} - \boldsymbol{w} - \boldsymbol{t}\|_2^2 \geq \|\boldsymbol{t}\|_2^2 \ , \quad \|\boldsymbol{x} + \boldsymbol{y} - \boldsymbol{w} - \boldsymbol{t}\|_2^2 \geq \|\boldsymbol{t}\|_2^2 \ .$$

(Otherwise, there would be a lattice vector closer to $\boldsymbol{t}$ than $\mathbf{0}$.) Rearranging as above, we have

$$\begin{aligned}
\delta_1 &:= \|\boldsymbol{w}\|_2^2 - 2\langle \boldsymbol{w}, \boldsymbol{t} \rangle \geq 0 \ , \\
\delta_2 &:= \|\boldsymbol{x} - \boldsymbol{w}\|_2^2 - \|\boldsymbol{x}\|_2^2 + 2\langle \boldsymbol{w}, \boldsymbol{t} \rangle = \|\boldsymbol{x} - \boldsymbol{w}\|_2^2 - 2\langle \boldsymbol{x}, \boldsymbol{t} \rangle + 2\langle \boldsymbol{w}, \boldsymbol{t} \rangle \geq 0 \ , \\
\delta_3 &:= \|\boldsymbol{y} - \boldsymbol{w}\|_2^2 - \|\boldsymbol{y}\|_2^2 + 2\langle \boldsymbol{w}, \boldsymbol{t} \rangle = \|\boldsymbol{y} - \boldsymbol{w}\|_2^2 - 2\langle \boldsymbol{y}, \boldsymbol{t} \rangle + 2\langle \boldsymbol{w}, \boldsymbol{t} \rangle \geq 0 \ , \\
\delta_4 &:= \|\boldsymbol{x} + \boldsymbol{y} - \boldsymbol{w}\|_2^2 - \|\boldsymbol{x} + \boldsymbol{y} - 2\boldsymbol{w}\|^2 - 2\langle \boldsymbol{w}, \boldsymbol{t} \rangle = \|\boldsymbol{x} + \boldsymbol{y} - \boldsymbol{w}\|_2^2 - 2\langle \boldsymbol{x}, \boldsymbol{t} \rangle - 2\langle \boldsymbol{y}, \boldsymbol{t} \rangle + 2\langle \boldsymbol{w}, \boldsymbol{t} \rangle \geq 0 \ ,
\end{aligned}$$

where we have used Eq. (6). Then,

$$\begin{aligned}
\delta_1 + \delta_2 + \delta_3 + \delta_4 &= \|\boldsymbol{w}\|_2^2 + \|\boldsymbol{x} - \boldsymbol{w}\|_2^2 + \|\boldsymbol{y} - \boldsymbol{w}\|_2^2 + \|\boldsymbol{x} + \boldsymbol{y} - \boldsymbol{w}\|_2^2 \\
&\quad - \|\boldsymbol{x}\|_2^2 - \|\boldsymbol{y}\|_2^2 - \|\boldsymbol{x} + \boldsymbol{y} - 2\boldsymbol{w}\|_2^2 \\
&= 0 \ .
\end{aligned}$$

Since the δ_i are all non-negative and they sum to zero, they must all be zero. In other words, $\boldsymbol{z}'_1, \boldsymbol{z}'_2, \boldsymbol{z}'_3, \boldsymbol{z}'_4 \in \mathrm{CVP}(\boldsymbol{t}, \mathcal{L})$ as needed.

Finally, notice that $2\boldsymbol{z}'_j = \boldsymbol{z}_1 + \boldsymbol{z}_2 + \boldsymbol{z}_3 + \boldsymbol{z}_4 - 2\boldsymbol{z}_j$. If $|C| < 8$, then there exists i, j such that $\boldsymbol{z}_i = \boldsymbol{z}'_j$. If $i \neq j$, then we see that $\boldsymbol{z}_i + \boldsymbol{z}_j = \boldsymbol{z}_k + \boldsymbol{z}_\ell$, i.e., the $\boldsymbol{z}_{i'}$ form a parallelogram. Furthermore, we must have $\boldsymbol{z}_j = \boldsymbol{z}'_i$, $\boldsymbol{z}_k = \boldsymbol{z}'_\ell$, and $\boldsymbol{z}_\ell = \boldsymbol{z}'_k$, i.e., $|C| = 4$. On the other hand, if $i = j$, then we have $4\boldsymbol{z}_i = \boldsymbol{z}_1 + \boldsymbol{z}_2 + \boldsymbol{z}_3 + \boldsymbol{z}_4$, which yields a contradiction because then $\boldsymbol{z}_i$ lies in the convex hull of the other vectors, which means that $\mathbf{B}\boldsymbol{z}_i$ cannot be distinct vectors equidistant from $\boldsymbol{t}$. □

The next two lemmas show some basic properties about the expressiveness of 3-SAT.

Lemma 6.9. *For any $k \geq 1$ and non-empty set $S \subseteq \{0,1\}^n$ with $|S| \leq 2^k$, there exists a k-CNF on n variables such that exactly $|S| - 1$ of the elements in S are satisfying assignments.*

Proof. We show how to find a k-clause that is satisfied by exactly $|S|-1$ elements. The proof is by induction on k. The base case $k=1$ is trivial. So, we suppose that the result holds for $k-1$. We assume without loss of generality that the number of strings in S whose first coordinate is one is between 1 and 2^{k-1}. (I.e., we assume that there are at least as many zeros as ones and that not all strings are the same on this coordinate.) Let S_1 be the set of strings with non-zero first coordinate. By induction, there is a $(k-1)$-clause ϕ such that exactly $|S_1|-1$ elements in S_1 satisfy ϕ. Then $\phi \vee \neg x_1$ is a k-clause satisfied by exactly $|S|-1$ elements in S, as needed. □

Lemma 6.10. *For any non-empty disjoint sets $S, T \subseteq \{0,1\}^n$ with $|S| = 4$ and $|T| \geq 2$, there exists a 3-CNF on n variables such that all elements in S are satisfying assignments and at least one element in T is not a satisfying assignment.*

Proof. We will find an assignment of 3 variables that satisfies S, but doesn't satisfy at least one element of T.

Define the majority string $s \in \{0,1\}^n$ of S to be such that $s_i = 0$, if for at least 2 strings in S, the i-th coordinate is 0, and $s_i = 1$, otherwise. Let $t \in T \setminus \{s\}$. Consider a position j where t differs from s. Set the j-th variable $x_j = s_j$. This satisfies at least 2 of the strings in S. Let a, b be the two strings in S such that $a_j = b_j \neq s_j$. Note that $t_j \neq s_j$, and hence $t_j = a_j = b_j$. Since t is different from a and b, there exist positions k and ℓ such that $t_k \neq a_k$ and $t_\ell \neq b_\ell$. We set $x_k = a_k$ and $x_\ell = b_\ell$. Thus, we satisfy every element of S but do not satisfy t. □

Finally, we prove Theorem 6.6. To do so, we first use Lemmas 6.8 and 6.10 to argue that if $\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \boldsymbol{z}_4 \in A$ satisfy $\boldsymbol{z}_1 + \boldsymbol{z}_2 + \boldsymbol{z}_3 + \boldsymbol{z}_4 = \mathbf{0} \bmod 2$ as in Lemma 6.8, then the $\boldsymbol{z}_i$ must form a parallelogram, where $A := f(\{0,1\}^n)$ is the image of f. We therefore conclude that if $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_8 \in A$ form an affine 3-cube modulo two, then they must actually form a parallelepiped. From this and Lemma 6.9, we derive a contradiction by Lemma 6.1. Therefore, $A \bmod 2$ cannot contain any affine 3-cube, which means that $n' > 4(n-2)/3$ by Lemma 6.7.

Proof of Theorem 6.6. Let R be a natural reduction from 3-SAT on n variables to CVP_2 on rank n' lattices. I.e., R maps 3-SAT instances ϕ to $\mathbf{B} \in \mathbb{R}^{d \times n'}$ and $\boldsymbol{t} \in \mathbb{R}^d$. First, notice that f must be injective. In particular, if f is not injective, then the reduction cannot possibly be valid because for every two distinct assignments $\boldsymbol{x}, \boldsymbol{x}' \in \{0,1\}^n$, there exists a 3-SAT instance ϕ that is satisfied by one but not the other. Let $A := f(\{0,1\}^n) \subset \mathbb{Z}^{n'}$ be the image of f.

Suppose that there exist distinct $\boldsymbol{z}_1 := f(\boldsymbol{x}_1), \boldsymbol{z}_2 := f(\boldsymbol{x}_2), \boldsymbol{z}_3 := f(\boldsymbol{x}_3), \boldsymbol{z}_4 := f(\boldsymbol{x}_4) = \boldsymbol{z}_1 + \boldsymbol{z}_2 + \boldsymbol{z}_3 - 2\boldsymbol{v} \in A$ for some $\boldsymbol{v} \in \mathbb{Z}^{n'}$. Then, for any $\mathbf{B}, \boldsymbol{t}$, if $\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \boldsymbol{z}_4 \in \mathrm{CVP}(\boldsymbol{t}, \mathbf{B})$, by Lemma 6.8, we must also have $\boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{z}_3', \boldsymbol{z}_4' \in\in \mathrm{CVP}(\boldsymbol{t}, \mathbf{B})$ as well, where

$$\boldsymbol{z}_1' := \boldsymbol{z}_2 + \boldsymbol{z}_3 - \boldsymbol{v}, \quad \boldsymbol{z}_2' := \boldsymbol{z}_1 + \boldsymbol{z}_3 - \boldsymbol{v}, \quad \boldsymbol{z}_3' := \boldsymbol{z}_1 + \boldsymbol{z}_2 - \boldsymbol{v}, \quad \boldsymbol{z}_4' := \boldsymbol{v}\ .$$

Therefore, by applying R to, e.g., the empty formula $\emptyset$, we see that $\boldsymbol{z}_1', \ldots, \boldsymbol{z}_4' \in A$ must also lie in the image of f, i.e., $\boldsymbol{z}_j' = f(\boldsymbol{x}_j')$. Again by Lemma 6.8, either the $\boldsymbol{z}_i$ form a parallelogram, or the sets $S := \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_4\}$ and $S' := \{\boldsymbol{x}_1', \ldots, \boldsymbol{x}_4'\}$ are disjoint. But, if S, S' are disjoint, then by Lemma 6.10, there exists a 3-clause ϕ such that $\phi(\boldsymbol{x}_i) = 1$ for all i but there exists a j such that $\phi(\boldsymbol{x}_j') = 0$. Then, taking $\mathbf{B}, \boldsymbol{t} = R(\phi)$, we see that $\boldsymbol{z}_j' \notin \mathrm{CVP}(\boldsymbol{t}, \mathbf{B})$, a contradiction.

We conclude that any such $\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \boldsymbol{z}_4 = \boldsymbol{z}_1 + \boldsymbol{z}_2 + \boldsymbol{z}_3 - 2\boldsymbol{v} \in A$ must form a parallelogram. I.e., if the $\boldsymbol{z}_i$ form an affine subspace mod two, then they form a parallelogram $\{\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \boldsymbol{z}_4\} =$

$\{\boldsymbol{y}_0, \boldsymbol{y}_0+\boldsymbol{y}_1, \boldsymbol{y}_0+\boldsymbol{y}_2, \boldsymbol{y}_0+\boldsymbol{y}_1+\boldsymbol{y}_2\}$. Now, suppose that A modulo two contains an affine 3-cube. I.e., suppose that it contains distinct $\boldsymbol{z}_1 := f(\boldsymbol{x}_1), \ldots, \boldsymbol{z}_8 := f(\boldsymbol{x}_8)$ such that $\boldsymbol{z}_i - \boldsymbol{y}_0 - \sum_{j\in W_i} \boldsymbol{y}_j \in 2\mathcal{L}$ for some distinct $W_i \subseteq \{1,2,3\}$. Then, by the above, we see that $\boldsymbol{z}_i = \boldsymbol{y}_0 + \sum_{j\in W_i} \boldsymbol{y}_j$. I.e., the $\boldsymbol{z}_i$ form a parallelepiped. But, by Lemma 6.9, there exists a 3-clause ϕ such that exactly seven out of the eight $\boldsymbol{x}_i$ satisfy ϕ. Therefore, $(\mathbf{B}, \boldsymbol{t}) := R(\phi)$ must have $\|\mathbf{B}\boldsymbol{z}_i - \boldsymbol{t}\| = \mathrm{dist}(\boldsymbol{t}, \mathbf{B}\mathbb{Z}^{n'})$ for seven out of the eight $\boldsymbol{z}_i$. But, by Lemma 6.1, this is not possible.

Finally, we conclude that A cannot include any affine 3-cube modulo two. Therefore, by Lemma 6.7, we see that $n' > 4(n-2)/3$. □

References

[A+19] Frank Arute et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779), 2019.

[AC19] Divesh Aggarwal and Eldon Chung. A note on the concrete hardness of the Shortest Independent Vectors Problem in lattices. 2019.

[ADS15] Divesh Aggarwal, Daniel Dadush, and Noah Stephens-Davidowitz. Solving the Closest Vector Problem in 2^n time— The discrete Gaussian strikes again! In *FOCS*, 2015.

[AM09] Per Austrin and Elchanan Mossel. Approximation resistant predicates from pairwise independence. *Computational Complexity*, 18(2):249–271, 2009.

[AM18] Divesh Aggarwal and Priyanka Mukhopadhyay. Faster algorithms for SVP and CVP in the ℓ_∞ norm. In *ISAAC*, 2018.

[AS18a] Divesh Aggarwal and Noah Stephens-Davidowitz. (Gap/S)ETH hardness of SVP. In *STOC*, 2018.

[AS18b] Divesh Aggarwal and Noah Stephens-Davidowitz. Just take the average! An embarrassingly simple 2^n-time algorithm for SVP (and CVP). In *SOSA*, 2018.

[BDGL16] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *SODA*, 2016.

[BGKM18] Arnab Bhattacharyya, Suprovat Ghoshal, Karthik C. S., and Pasin Manurangsi. Parameterized Intractability of Even Set and Shortest Vector Problem from Gap-ETH. In *ICALP*, 2018.

[BGS17] Huck Bennett, Alexander Golovnev, and Noah Stephens-Davidowitz. On the quantitative hardness of CVP. In *FOCS*, 2017.

[BN09] Johannes Blömer and Stefanie Naewe. Sampling methods for shortest vectors, closest vectors and successive minima. *Theoret. Comput. Sci.*, 410(18):1648–1665, 2009.

[Cha16] Siu On Chan. Approximation resistance from pairwise-independent subgroups. *J. ACM*, 63(3):27:1–27:32, 2016.

[CS16] Gil Cohen and Igor Shinkar. The complexity of DNF of parities. In *ITCS*, pages 47–58. ACM, 2016.

[Dad12] Daniel Dadush. A $O(1/\varepsilon^2)^n$-time sieving algorithm for approximate Integer Programming. In *LATIN*, 2012.

[DKRS03] Irit Dinur, Guy Kindler, Ran Raz, and Shmuel Safra. Approximating CVP to within almost-polynomial factors is NP-hard. *Combinatorica*, 23(2):205–243, 2003.

[GMSS99] Oded Goldreich, Daniele Micciancio, Shmuel Safra, and Jean-Pierre Seifert. Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. *Inf. Process. Lett.*, 71(2):55 – 61, 1999.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[Kan87] Ravi Kannan. Minkowski's convex body theorem and Integer Programming. *Math. Oper. Res.*, 12(3):415–440, 1987.

[Man19] Pasin Manurangsi. *Approximation and Hardness: Beyond P and NP*. PhD thesis, University of California, Berkeley, 2019.

[MM17] Konstantin Makarychev and Yury Makarychev. Approximation algorithms for csps. In *The Constraint Satisfaction Problem: Complexity and Approximability*, pages 287–325. 2017.

[MV13] Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. *SIAM J. Comput.*, 42(3):1364–1391, 2013.

[NIS16] NIST post-quantum standardization call for proposals. http://csrc.nist.gov/groups/ST/post-quantum-crypto/cfp-announce-dec2016.html, 2016. Accessed: 2017-04-02.

[RR06] Oded Regev and Ricky Rosen. Lattice problems and norm embeddings. In *STOC*, 2006.

[SV19] Noah Stephens-Davidowitz and Vinod Vaikuntanathan. SETH-hardness of coding problems. In *FOCS*, 2019.

[vEB81] Peter van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical report, 8104, University of Amsterdam, Department of Mathematics, Netherlands, 1981.

[Wil05] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005.

A Hardness of SVP

We notice that Theorem 1.1, or more specifically Corollary 3.12, immediately implies an improvement to the main result in [AS18a]. Specifically, while [AS18a, Theorem 4.3] previously only applied to some non-explicit set of p, we can now extend it to all $p \gtrsim 2.14$ with $p \notin 2\mathbb{Z}$.

We give the formal statement below for completeness. The proof is essentially identical to the original. We simply substitute our Corollary 3.12 for the main result from [BGS17] (noting, as

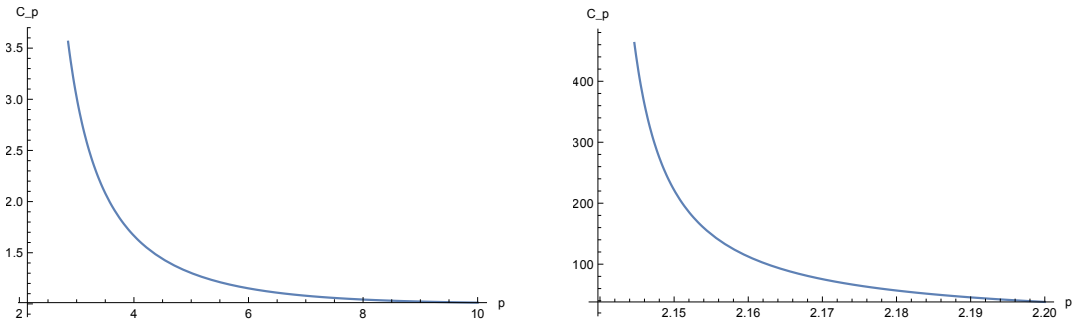

Figure 3: The value C_p for different values of $p > p_0$. In particular, for $p \notin 2\mathbb{Z}$, there is no $2^{n/C_p}$-time algorithm for SVP_p unless SETH is false. The plot on the left shows C_p over a wide range of p, while the plot on the right shows the behavior when p is close to the threshold $p_0 \approx 2.13972$. (Figure taken from [AS18a].)

in [AS18a] that the hard CVP_p instance promised by Corollary 3.12 has a particularly nice form). We also include a plot of C_p in Figure 3, which is taken from [AS18a]. ([AS18a] also proved that there is no $2^{o(n)}$-time algorithm for SVP_p for any p assuming Gap-ETH.)

Theorem A.1. *For any integer $k \geq 2$ and $p > p_0$ with $p \notin 2\mathbb{Z}$, there is an efficient randomized reduction from Max-k-SAT on n variables to SVP_p on a lattice of rank $\lceil C_p n + \log^2 n \rceil$, where*

$$C_p := \frac{1}{1 - \log_2 W_p} \qquad \text{and} \qquad W_p := \min_{\tau > 0} \exp(\tau/2^p)\Theta_p(\tau) \; .$$

Here, $\Theta_p(\tau) := \sum_{z \in \mathbb{Z}} \exp(-\tau |z|^p)$, and $p_0 \approx 2.13972$ is the unique solution to the equation $W_{p_0} = 2$.

In particular, for every $\varepsilon > 0$ and $p > p_0$ with $p \notin 2\mathbb{Z}$ there is no $2^{(1-\varepsilon)n/C_p}$-time algorithm for CVP_p unless SETH is false.