Noname manuscript No.
(will be inserted by the editor)

A Randomized Sieving Algorithm for Approximate Integer Programming

Daniel Dadush

the date of receipt and acceptance should be inserted later

Abstract The Integer Programming Problem (IP) for a polytope $P \subseteq \mathbb{R}^n$ is to find an integer point in P or decide that P is integer free. We give a randomized algorithm for an approximate version of this problem, which correctly decides whether P contains an integer point or whether a $(1+\epsilon)$-scaling of P about its center of gravity is integer free in $2^{O(n)}(1/\epsilon^2)^n$-time and $2^{O(n)}(1/\epsilon)^n$-space with overwhelming probability. Our algorithm proceeds by reducing the approximate IP problem to an approximate Closest Vector Problem (CVP) under a "near-symmetric" norm. Our main technical contribution is an extension of the AKS randomized sieving technique, first developed by Ajtai, Kumar, Sivakumar (STOC 2001) for lattice problems under the ℓ_2 norm, to the setting of asymmetric norms. We also present an application of our techniques to exact IP, where we give a nearly optimal algorithmic implementation of the Flatness Theorem, a central ingredient for many IP algorithms. Our results also extend to general convex bodies and lattices.

Keywords Integer Programming · Lattice Problems · Shortest Vector Problem · Closest Vector Problem

1 Introduction

The Integer Programming (IP) Problem, i.e. the problem of deciding whether a polytope contains an integer point, is a classic problem in Operations Research and Computer Science. Algorithms for IP were first developed in the 1950s when Gomory [1] gave a finite cutting plane algorithm to solve general (Mixed)-Integer Programs. However, the first algorithms with complexity guarantees (i.e. better than finiteness) came much later. The first such algorithm was the breakthrough result of Lenstra [2], which gave the first fixed dimension

Daniel Dadush
Computer Science Department, New York University, 251 Mercer Street, New York, NY 10012, USA. E-mail: dadush@cs.nyu.edu

polynomial time algorithm for IP. Lenstra's approach revolved on finding "flat" integer directions of a polytope, and achieved a leading complexity term of $2^{O(n^3)}$ where n is the number of variables. Lenstra's approach was generalized and substantially improved upon by Kannan [3], who used an entire short lattice basis to yield an $O(n^{2.5})^n$-time and poly(n)-space algorithm. In [4], Hildebrand and Köppe use strong ellipsoidal rounding and a recent solver for the Shortest Vector Problem (SVP) under the ℓ_2 norm [5] to give an $2^{O(n)}n^{2n}$-time and $2^{O(n)}$-space algorithm for IP. Lastly, Dadush et al [6] use a solver for SVP under general norms to give a $2^{O(n)}(n^{\frac{4}{3}}\text{ polylog}(n))^n$-expected time and $2^{O(n)}$-space algorithm. Following the works of Lenstra and Kannan, fixed dimension polynomial algorithms were discovered for many related problems such as counting the number of integer points in a rational polyhedron [7], parametric integer programming [8,9], and integer optimization over quasi-convex polynomials [10,4]. However, over the last twenty years the known algorithmic complexity of IP has only modestly decreased. A central open problem in the area therefore remains the following [5,4,6]:

Problem 1 Does there exist a $2^{O(n)}$-time algorithm for Integer Programming?

In this paper, we show that if one is willing to accept an approximate notion of containment then the answer to the above question is affirmative. More precisely, we give a randomized algorithm which can correctly distinguish whether a polytope P contains an integer point or if a $(1+\epsilon)$-dilation of P about its center of gravity contains no integer points in $2^{O(n)}(1/\epsilon^2)^n$-time and $2^{O(n)}(1/\epsilon)^n$-space with overwhelming probability. Our results naturally extend to the setting of general convex bodies and lattices, where the IP problem in this context is to decide for a convex body K and lattice $\mathcal{L}$ in $\mathbb{R}^n$ whether $K \cap \mathcal{L} \neq \emptyset$. To obtain the approximate IP result, we reduce the problem to a $(1+\epsilon)$-approximate Closest Vector Problem (CVP) under a "near-symmetric" norm.

Given an n-dimensional lattice $\mathcal{L} \subseteq \mathbb{R}^n$ (integer combinations of a basis $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^n$) the SVP is to find $\min_{\mathbf{y} \in \mathcal{L}\setminus\{0\}} \|\mathbf{y}\|$, and given $\mathbf{x} \in \mathbb{R}^n$ the CVP is to find $\min_{\mathbf{y}\in\mathcal{L}} \|\mathbf{y} - \mathbf{x}\|$, where $\|\cdot\|$ is a given (asymmetric-)norm. An asymmetric norm $\|\cdot\|$ satisfies all the standard norm properties except symmetry, i.e. we allow $\|\mathbf{x}\| \neq \|-\mathbf{x}\|$.

Our methods in this setting are based on a randomized sieving technique first developed by Ajtai, Kumar and Sivakumar [11,12] for solving the Shortest (SVP) and approximate Closest Vector Problem (CVP). In [11], they give a $2^{O(n)}$-time and space randomized sieving algorithm for SVP in the ℓ_2 norm, extending this in [12] to give a $2^{O(\frac{n}{\epsilon})}$-time and space randomized algorithm for $(1+\epsilon)$-CVP in the ℓ_2 norm. In [13], Blomer and Naewe adapt the AKS sieve to give a $2^{O(n)}$-time and space randomized algorithm for ℓ_p SVP, and a $2^{O(n)}(1/\epsilon^2)^n$-time and $2^{O(n)}(1/\epsilon)^n$-space randomized algorithm for $(1+\epsilon)$-CVP under ℓ_p norms. In [14], the previous results are extended to give a $2^{O(n)}$-time and space randomized algorithm for the SVP under any symmetric norm. In [15], a technique to boost any 2-approximation algorithm for ℓ_∞ CVP is

given which yields a $2^{O(n)}(\ln(\frac{1}{\epsilon}))^n$ and $2^{O(n)}$-space algorithm for $(1+\epsilon)$-CVP under ℓ_∞.

Our main technical contribution is an extension of the AKS sieving technique to give a $2^{O(n)}(1/\epsilon^2)^n$-time and $2^{O(n)}(1/\epsilon)^n$-space randomized algorithm for CVP under *any near-symmetric norm*.

1.1 Definitions

In what follows, $K \subseteq \mathbb{R}^n$ will denote a convex body (a full dimensional compact convex set) and $\mathcal{L} \subseteq \mathbb{R}^n$ will denote an n-dimensional lattice. We define the *dual lattice* of $\mathcal{L}$ as $\mathcal{L}^* = \{\mathbf{y} \in \mathbb{R}^n : \langle \mathbf{y}, \mathbf{x} \rangle \in \mathbb{Z} \ \ \forall \mathbf{x} \in \mathcal{L}\}$. K will be presented by a *membership oracle* in the standard way (see section 2), and $\mathcal{L}$ will be presented by a generating basis $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^n$. We define the *barycenter* (or centroid) of K as $\mathbf{b}(K) = \frac{1}{\mathrm{vol}_n(K)} \int_K \mathbf{x} d\mathbf{x}$.

For sets $A, B \subseteq \mathbb{R}^n$ and scalars $s, t \in \mathbb{R}$ define the *Minkowski sum* $sA+tB = \{s\mathbf{a} + t\mathbf{b} : \mathbf{a} \in A, \mathbf{b} \in B\}$. $\mathrm{int}(A)$ denotes the interior of the set A.

Let $C \subseteq \mathbb{R}^n$ be a convex body where $0 \in \mathrm{int}(C)$. Define the norm (possibly asymmetric) induced by C (or *gauge function* of C) as $\|\mathbf{x}\|_C = \inf\{s \geq 0 : \mathbf{x} \in sC\}$ for $\mathbf{x} \in \mathbb{R}^n$. $\|\cdot\|_C$ satisfies all standard norm properties except symmetry, i.e. $\|\mathbf{x}\|_C \neq \|-\mathbf{x}\|_C$ is allowed. $\|\cdot\|_C$ (or C) is γ-*symmetric*, for $0 < \gamma \leq 1$, if $\mathrm{vol}_n(C \cap -C) \geq \gamma^n \mathrm{vol}_n(C)$. Note C is 1-symmetric iff $C = -C$. For the sake of concision, we shall now use the generic term norm to include both asymmetric and symmetric norms.

For a lattice $\mathcal{L}$ and norm $\|\cdot\|_C$, define the *first minimum* of $\mathcal{L}$ under $\|\cdot\|_C$ as $\lambda_1(C, \mathcal{L}) = \inf_{\mathbf{z} \in \mathcal{L} \setminus \{0\}} \|\mathbf{z}\|_C$ (length of shortest non-zero vector). For a target $\mathbf{x}$, lattice $\mathcal{L}$, and norm $\|\cdot\|_C$, define the distance from $\mathbf{x}$ to $\mathcal{L}$ under $\|\cdot\|_C$ as $d_C(\mathcal{L}, \mathbf{x}) = \inf_{\mathbf{z} \in \mathcal{L}} \|\mathbf{z} - \mathbf{x}\|_C$.

1.2 Results

We state our main result in terms of general convex bodies and lattices. We recover the standard integer programming setting by setting $\mathcal{L} = \mathbb{Z}^n$, the standard integer lattice, and $K = \{\mathbf{x} \in \mathbb{R}^n : A\mathbf{x} \leq \mathbf{b}\}$, a general polytope. For simplicity, we often omit standard polynomial factors from the runtimes of our algorithms (i.e. polylog terms associated with bounds on K or the bit length of the basis for $\mathcal{L}$).

Our main result is the following:

Theorem 1 (Approximate IP Feasibility) *For $0 < \epsilon \leq \frac{1}{2}$, there exists a $2^{O(n)}(1/\epsilon^2)^n$-time and $2^{O(n)}(1/\epsilon)^n$-space randomized algorithm which with probability at least $1 - 2^{-n}$ either outputs a point*

$$\mathbf{y} \in ((1+\epsilon)K - \epsilon\mathbf{b}(K)) \cap \mathcal{L}$$

or correctly decides that $K \cap \mathcal{L} = \emptyset$*. Furthermore, if*

$$\left(\frac{1}{1+\epsilon}K + \frac{\epsilon}{1+\epsilon}\mathbf{b}(K)\right) \cap \mathcal{L} \neq \emptyset,$$

the algorithm returns a point $z \in K \cap \mathcal{L}$ *with probability at least* $1 - 2^{-n}$*.*

Here we note that $((1+\epsilon)K - \epsilon\mathbf{b}(K)) = (1+\epsilon)(K - \mathbf{b}(K)) + \mathbf{b}(K)$, corresponds to a $(1+\epsilon)$-scaling of K about its barycenter $\mathbf{b}(K)$.

The above theorem shows that IP can be solved much faster than the current $n^{O(n)}$ worst case running time when K contains a "deep" lattice point (i.e. within a slight scaling of K around its barycenter). Indeed, as long as

$$\left(\frac{1}{1+n^{-1/2}}K + \frac{n^{-1/2}}{1+n^{-1/2}}\mathbf{b}(K)\right) \cap \mathcal{L} \neq \emptyset,$$

our algorithm will find an integer point in $2^{O(n)}n^n$-time (which is the conjectured complexity of the IP algorithm in [6]). Hence to improve the time complexity of IP below $2^{O(n)}n^{\delta n}$, for any $0 < \delta < 1$, one may assume that all the integer points lie close to the boundary, i.e. that

$$\left(\frac{1}{1+n^{-\frac{1}{2}\delta}}K + \frac{n^{-\frac{1}{2}\delta}}{1+n^{-\frac{1}{2}\delta}}\mathbf{b}(K)\right) \cap \mathcal{L} = \emptyset.$$

The above statement helps elucidate the structure of the "hard" IP instances for current algorithms, i.e. they are instances with many infeasible lattice points lying near the boundary of K.

Starting from the above algorithm, we can use a binary search procedure to go from approximate feasibility to approximate optimization. This yields the following theorem:

Theorem 2 (Approximate Integer Optimization) *For* $\mathbf{v} \in \mathbb{R}^n$*,* $0 < \epsilon \leq \frac{1}{2}$*,* $\delta > 0$*, there exists a* $2^{O(n)}(1/\epsilon^2)^n \operatorname{polylog}(\frac{1}{\delta}, \|\mathbf{v}\|_2)$*-time and* $2^{O(n)}(1/\epsilon)^n$*-space randomized algorithm which with probability at least* $1 - 2^{-n}$ *either outputs*

$$\mathbf{y} \in (K + \epsilon(K - K)) \cap \mathcal{L}$$

such that

$$\sup_{\mathbf{z} \in K \cap \mathcal{L}} \langle \mathbf{v}, \mathbf{z} \rangle \leq \langle \mathbf{v}, \mathbf{y} \rangle + \delta$$

or correctly decides that $K \cap \mathcal{L} = \emptyset$*.*

The above theorem states that if we wish to optimize over $K \cap \mathcal{L}$, we can find a lattice point in a slight blowup of K whose objective value is essentially as good as any point in $K \cap \mathcal{L}$. We remark that the blowup is worse than in Theorem 1, since $(1+\epsilon)K - \epsilon\mathbf{x} \subseteq K + \epsilon(K - K)$ for any $\mathbf{x} \in K$. This stems from the need to call the feasibility algorithm on multiple restrictions of K. For a vector $\mathbf{v} \in \mathbb{R}^n$, we define the *width* of K with respect to $\mathbf{v}$ as

$$\operatorname{width}_K(\mathbf{v}) = \sup_{\mathbf{x} \in K} \langle \mathbf{x}, \mathbf{v} \rangle - \inf_{\mathbf{x} \in K} \langle \mathbf{x}, \mathbf{v} \rangle .$$

It is easy to check that $\text{width}_K(\cdot)$ defines a symmetric norm on $\mathbb{R}^n$. To get a clearer understanding of the "blowup body", the new constraints of $K+\epsilon(K-K)$ can be understood from the following formula:

$$\sup_{\mathbf{x}\in K+\epsilon(K-K)} \langle \mathbf{v}, \mathbf{x}\rangle = \sup_{\mathbf{x}\in K} \langle \mathbf{v}, \mathbf{x}\rangle + \epsilon \text{width}_K(\mathbf{v}).$$

Hence each valid constraint $\langle \mathbf{v}, \mathbf{x}\rangle \leq c$ for K, is relaxed by an ϵ-fraction of $\mathbf{v}$'s width (or variation) with respect to K.

Application to Exact Integer Programming. A natural question is whether the approximate IP problem has a direct application to exact IP. While we have not yet found a natural class of IPs for which our approximate IP solver gives provably optimal or near-optimal solutions (since we do not know how to control which constraints are violated), we can instead show the utility of our solver as a tool for solving the exact problem. In his breakthrough work, Lenstra [2] showed how to efficiently solve IPs using small lattice width directions. At the core of Lenstra's algorithm is a subroutine that given a convex body K[1] and lattice $\mathcal{L}$, either outputs a lattice point in K, or outputs a decomposition of $\mathcal{L}$ into consecutive parallel lattice hyperplanes such that only a "small" number of these hyperplanes intersect K. To solve the IP when the latter case occurs, the idea is to recursively solve the $n-1$ dimensional IPs indexed by the lattice hyperplanes intersecting K (since any lattice point in K must be contained in exactly one of them), and return any lattice point found by the recursive calls. This strategy still forms the basis of the fastest current IP algorithms [4,6], where the bulk of the new work has focused on finding better hyperplane decompositions (i.e. decompositions minimizing the number hyperplanes intersecting K).

For the main application of our approximate IP solver, we use it to give an algorithm which either outputs a lattice point inside K, or outputs a nearly "optimal" hyperplane decomposition with respect to K. To explain what drives the "thinness" of these decompositions (i.e. the number of intersecting hyperplanes), we must first describe Kinchine's Flatness theorem in the geometry of numbers. We define the *lattice width* of K with respect to $\mathcal{L}$ as

$$\text{width}(K, \mathcal{L}) = \min_{\mathbf{y}\in\mathcal{L}^*\setminus\{\mathbf{0}\}} \text{width}_K(\mathbf{y}) \tag{1}$$

We note that finding a smallest width non-zero dual vector corresponds exactly to an SVP on $\mathcal{L}^*$ with respect to the symmetric norm $\text{width}_K(\cdot)$. The lattice width essentially corresponds to the optimal thinness of any hyperplane decomposition of $\mathcal{L}$ with respect to K. To see this, take $\mathbf{y} \in \mathcal{L}^* \setminus \{\mathbf{0}\}$ such that $\text{width}_K(\mathbf{y}) = \text{width}(K, \mathcal{L})$, and let $H^i_{\mathbf{y}} = \{\mathbf{x} : \langle \mathbf{x}, \mathbf{y}\rangle = i\}$, for $i \in \mathbb{Z}$. Since $\mathbf{y} \neq \mathbf{0}$, we see that the $H^i_{\mathbf{y}}$ are distinct parallel hyperplanes, and since $\mathbf{y} \in \mathcal{L}^*$

[1] This was originally done for K a polyhedron, though this makes no essential difference.

we have that each $H^i_{\mathbf{y}}$ corresponds to the affine hull of the lattice points it contains (i.e. a lattice hyperplane) and that

$$\mathcal{L} \subseteq \bigcup_{i\in\mathbb{Z}} H^i_{\mathbf{y}} \Rightarrow K \cap \mathcal{L} \subseteq \bigcup_{i\in\mathbb{Z}: H^i_{\mathbf{y}}\cap K\neq\emptyset} H^i_{\mathbf{y}}.$$

Furthermore, by convexity of K the set of hyperplanes $H^i_{\mathbf{y}}$ intersecting K are consecutive, i.e. $\{i \in \mathbb{Z} : H^i_{\mathbf{y}} \cap K \neq \emptyset\}$ is an interval in $\mathbb{Z}$. Lastly, it is not hard to check that $\lfloor \text{width}_K(\mathbf{y})\rfloor \leq |\{i \in \mathbb{Z} : H^i_{\mathbf{y}} \cap K \neq \emptyset\}| \leq \lfloor \text{width}_K(\mathbf{y})\rfloor + 1$. In conclusion, if K has $\text{width}_K(\mathcal{L}) = \lambda$, then the optimal hyperplane decomposition of $\mathcal{L}$ with respect to K has thinness between $\lfloor\lambda\rfloor$ and $\lfloor\lambda\rfloor + 1$.

Clearly, without any assumptions on K and $\mathcal{L}$, $\mathcal{L}$ need not admit a thin lattice decomposition with respect to K (indeed $\text{width}(K, \mathcal{L})$ may be arbitrarily large). Therefore, from the perspective of IP, it is important to have a good sufficient condition for the lattice width to be small. The Flatness theorem, first proven by Kinchine [16], states that if K is lattice-free (i.e. K contains no lattice points), then K has lattice width bounded by a function of dimension alone. We define the flatness constant of K as

$$\text{Fl}(K) = \sup\{\text{width}(TK, \mathbb{Z}^n) : T \text{ affine invertible }, TK \cap \mathbb{Z}^n = \emptyset\} \quad (2)$$

Also, for $n \in \mathbb{N}$, we define the n-dimensional flatness constant

$$\text{Fl}(n) = \sup\{\text{Fl}(K) : \text{ convex body } K \subseteq \mathbb{R}^n\},$$

i.e. the worst case flatness bound over all n dimensional convex bodies. From the definition, we see that $\text{Fl}(K)$ is invariant under invertible affine transformations of K. Furthermore, it is also easy to check that one can replace $\mathbb{Z}^n$ in the definition with any other lattice without changing $\text{Fl}(K)$ (since all n-dimensional lattices are related by linear transformation). Given the importance to IP, much work has focused on the problem of deriving strong bounds for the flatness constant [17–23]. We provide the best known bounds below:

Theorem 3 (Flatness Theorem) *For a convex body $K \subseteq \mathbb{R}^n$,*

- *[21]* $\text{Fl}(K) = \Omega(n)$.
- *[23]* $\text{Fl}(K) = O(n^{\frac{4}{3}}\,\text{polylog}(n))$.
- *[22] if K is a polytope with $O(n^c)$ facets,* $\text{Fl}(K) = O(cn\log n)$.
- *[20] if K is an ellipsoid,* $\text{Fl}(K) = O(n)$.

We now present the main result of this section:

Theorem 4 (Algorithmic Flatness Theorem) *For $0 \leq \epsilon \leq \frac{1}{2}$, there exists a $2^{O(n)}(\frac{1}{\epsilon^2})^n$-time and $2^{O(n)}(\frac{1}{\epsilon})^n$-space randomized algorithm that correctly outputs one of the following with probability at least $1 - 2^{-n}$:*

(a) EMPTY if $K \cap \mathcal{L} = \emptyset$ (guaranteed if $((1+\epsilon)K - \epsilon\mathbf{b}(K)) \cap \mathcal{L} = \emptyset$), or
(b) $\mathbf{z} \in K \cap \mathcal{L}$ (guaranteed if $\left(\frac{1}{1+\epsilon}K + \frac{\epsilon}{1+\epsilon}\mathbf{b}(K)\right) \cap \mathcal{L} \neq \emptyset$), or
(c) $\mathbf{y} \in \mathcal{L}^* \setminus \{\mathbf{0}\}$ satisfying $\text{width}_K(\mathbf{y}) = \text{width}(K, \mathcal{L}) \leq (1+\epsilon)\text{Fl}(K)$.

In the above theorem, the guarantees in (a)-(c) should be interpreted as follows: *conditioned* on the "success" of the algorithm (which occurs with probability $1 - 2^{-n}$), and assuming that the condition in the guarantee is satisfied, then the stated outcome will occur with conditional probability 1. Note that this does not imply that a certain outcome cannot occur if the condition from the guarantee is violated. For example, it is possible that the algorithm correctly returns a point $\mathbf{y} \in K \cap \mathcal{L}$ even if $\mathbf{y} \in \partial K$ (and hence not "deep inside" K).

Given our current toolset, implementing the above algorithm is straightforward. First, we run the approximate IP solver on K with parameter ϵ. Now we need only determine what to do if the solver neither decides that $K \cap \mathcal{L} = \emptyset$, nor returns a lattice point $\mathbf{x} \in K \cap \mathcal{L}$. In this case, we run any general norm SVP solver [14,6] to compute a shortest non-zero vector in $\mathcal{L}^*$ under the norm $\text{width}_K(\cdot)$. Here we get the desired bound on $\text{width}(K, \mathcal{L})$ from the fact that K is nearly lattice-free.

The above algorithm can be thought of as a nearly "optimal" implementation of the Flatness theorem. Comparing to previous algorithms with guarantees of the above type, the previous best obtained hyperplane decompositions of thinness $O(n^2)$ [4]. Hence here we gain a factor of at least $n^2/Fl(K) = \tilde{\Omega}(n^{\frac{2}{3}})$ (using Theorem 3). The blowup in most previous approaches stems from the need to use an ellipsoidal approximation of K, which generally results in an n factor blowup with respect to the ellipsoidal estimates. We note that using the above algorithm as the core IP subroutine (setting $\epsilon = \frac{1}{2}$), one directly gets a randomized $2^{O(n)}\text{Fl}(n)^n$-time and $2^{O(n)}$-space algorithm for exact IP (matching the complexity of [6]). To see this, note that any recursion node we create at most $(1+\epsilon)\text{Fl}(n) \leq 2\text{Fl}(n)$ subproblems, and hence the entire recursion tree has size at most $2^n\text{Fl}(n)^n$ (with very high probability). This controls the complexity of the algorithm since we never do more than $2^{O(n)}$ work at any recursion node.

In [6], they give an algorithm which constructs hyperplane decompositions of thinness $\tilde{O}(n^{\frac{4}{3}})$ in a slightly different manner than presented above. Here, they first compute $\text{width}_K(\mathcal{L})$ using a general norm SVP solver, and if $\text{width}_K(\mathcal{L}) \geq \text{Fl}(n)$, they replace K by $K' = \alpha K + (1-\alpha)\mathbf{x}$ (a scaled down version of K), for any $\mathbf{x} \in K$, satisfying $\text{width}_{K'}(n) = \text{Fl}(n)$. Since K' always has lattice width $\geq \text{Fl(n)}$, it is guaranteed that $K' \cap \mathcal{L} \neq \emptyset \Leftrightarrow K \cap \mathcal{L} \neq \emptyset$, and hence it suffices to solve the IP on K'. Furthermore, by definition K' now admits a hyperplane decomposition of thinness $O(\text{Fl}(n))$. The drawback of this approach is that it requires an explicit bound on the worst case flatness bound (our algorithm is agnostic to $\text{Fl}(n)$), it cannot guarantee near optimal thinness of the hyperplane decompositions on a per instance basis, and it provides very little information on how close K is to being lattice-free.

Unfortunately, it is unclear at this point whether using the algorithm from Theorem 4 can by itself enable an improvement in the worst case complexity of IP achieved in [6]. The reason for this is that in the worst case, the thinness of the hyperplane decompositions produced by both approaches remains the

essentially the same (though one would expect the average case to be much better for the Algorithm 4). However, with a more careful analysis, one might be able to show that using Algorithm 4 allows one to quickly prune a very large portion of the search tree. To justify this, we note that we only enter case (c) when K is both close to being lattice feasible and infeasible. Therefore, if very little pruning occurs during the IP algorithm, it means that almost all recursion nodes correspond to slices of K that lie on the boundary of lattice feasibility, a scenario which seems quite unlikely. We therefore believe that Algorithm 4 should, at the very least, provide a useful tool for future IP algorithms.

1.3 Main Tool

We now describe the main tool used to derive all of the above algorithms. At the heart of Theorem 1, is the following algorithm:

Theorem 5 *Let $\|\cdot\|_C$ denote a γ-symmetric norm. For $\mathbf{x} \in \mathbb{R}^n$, $0 < \epsilon \leq \frac{1}{2}$, there exists an $2^{O(n)}(\frac{1}{\gamma^4\epsilon^2})^n$-time and $2^{O(n)}(\frac{1}{\gamma^2\epsilon})^n$-space randomized algorithm which computes a point $\mathbf{y} \in \mathcal{L}$ satisfying*

$$\|\mathbf{y} - \mathbf{x}\|_C \leq (1+\epsilon) d_C(\mathcal{L}, \mathbf{x})$$

with probability at least $1 - 2^{-n}$. Furthermore, if $d_C(\mathcal{L}, \mathbf{x}) \leq t\lambda_1(C, \mathcal{L})$, for $t \geq 2$, then an exact closest vector can be found in randomized $2^{O(n)}(\frac{t^2}{\gamma^4})^n$-time and $2^{O(n)}(\frac{t}{\gamma^2})^n$-space with probability at least $1 - 2^{-n}$.

The above algorithm generalizes the AKS sieve to work for asymmetric norms. As mentioned previously [13] gave the above result for ℓ_p norms, and [14] gave a $2^{O(n)}$-time exact SVP solver for symmetric norms (also implied by the above since SVP $\leq$ CVP, see [24]). In [6], a Las Vegas algorithm (where only the runtime is probabilistic, not the correctness) is given for the exact versions of the above results (i.e. where an exact closest / shortest vector is found) with similar asymptotic complexity, which crucially uses the techniques of [5] developed for ℓ_2-CVP. In [5], Micciancio and Voulgaris give deterministic $2^{O(n)}$-time and space algorithms for both ℓ_2 SVP and CVP based upon Voronoi cell computations.

Hence compared with previous results, the novelty of the above algorithm is the extension of the AKS sieving technique for $(1+\epsilon)$-CVP under asymmetric norms. As seen from Theorems 1, 2, and 4 the significance of this extension is in its applications to IP. We note that asymmetry arises naturally in the context of IP, since the continuous relaxation of an IP need not be symmetric (or even closely approximable by a symmetric set). Furthermore, we believe our results illustrate the versatility of the AKS sieving paradigm.

From a high level, our algorithm uses the same framework as [13,14]. We first show that the AKS sieve can be used to solve the Subspace Avoiding Problem (SAP), which was first defined in [13], and use a reduction from CVP to SAP to get the final result. The technical challenge we overcome, is finding

the correct generalizations of the each of the steps performed in previous algorithms to the asymmetric setting. We discuss this further in section 3.2.

1.4 Organization

In section 2, we give some general background in convex geometry and lattices. In section 3.1, we describe the reductions from Approximate Integer Programming to Approximate CVP, Approximate Integer Optimization to Approximate Integer Programming, as well as our implementation of the Flatness theorem. In section 3.2, we present the algorithm for the Subspace Avoiding Problem, and in section 3.3 we give the reduction from CVP to SAP.

2 Preliminaries

Convexity: For a convex body $K \subseteq \mathbb{R}^n$, containing $\mathbf{0} \in \text{int}(K)$, we define the *polar* of K as $K^* = \{\mathbf{x} \in \mathbb{R}^n : \langle \mathbf{x}, \mathbf{y} \rangle \leq 1 \ \ \forall \mathbf{y} \in K\}$. By classical duality, we have that $(K^*)^* = K$. Furthermore, one can check that for $\mathbf{x} \in \mathbb{R}^n$, $\|\mathbf{x}\|_K = \inf\{s \geq 0 : \mathbf{x} \in sK\} = \sup_{\mathbf{y} \in K^*} \langle \mathbf{x}, \mathbf{y} \rangle$.

Computation Model: A convex body $K \subseteq \mathbb{R}^n$ is $(\mathbf{a}_0, r, R)$-centered if $\mathbf{a}_0 + rB_2^n \subseteq K \subseteq \mathbf{a}_0 + RB_2^n$, where B_2^n is the unit Euclidean ball. All the convex bodies in this paper will be $(\mathbf{a}_0, r, R)$-centered unless otherwise specified. To interact with K, algorithms are given access to a *membership oracle* for K, i.e. an oracle O_K such that $O_K(x) = 1$ if $\mathbf{x} \in K$ and 0. In some situations, an exact membership oracle is difficult to implement (e.g. deciding whether a matrix A has operator norm ≤ 1), in which situation we settle for a "weak"-membership oracle, which only guarantees its answer for points that are either ϵ-deep inside K or ϵ-far from K (the error tolerance ϵ is provided as an input to the oracle). We note that for a centered convex body $K \subseteq \mathbb{R}^n$ equipped with a weak membership oracle, one can approximately optimize any linear function over K in polynomial time (see [25] section 4.3 for additional details).

For a $(0, r, R)$-centered K the gauge function $\|\cdot\|_K$ is an asymmetric norm. To interact with norms, algorithms are given a distance oracle, i.e. a function which on input $\mathbf{x}$ returns $\|\mathbf{x}\|_K$. It is not hard to check that given a membership oracle for K, one can compute $\|\mathbf{x}\|_K$ to within any desired accuracy using binary search. Also we remember that $\|\mathbf{x}\|_K \leq 1 \Leftrightarrow \mathbf{x} \in K$, hence a distance oracle can easily implement a membership oracle. All the algorithms in this paper can be made to work with weak-oracles, but for simplicity in presentation, we assume that our oracles are all exact and that the conversion between different types of oracles occurs automatically. We note that when K is a polytope, all the necessary oracles can be implemented exactly and without difficulty.

In the oracle model of computation, complexity is measured by the number of oracles calls and arithmetic operations.

Probability: For random variables $X, Y \in \Omega$, we define the *total variation distance* between X and Y as

$$d_{TV}(X,Y) = \sup_{A \subseteq \Omega} |\Pr(X \in A) - \Pr(Y \in A)|$$

The following lemma is a standard fact in probability theory:

Lemma 1 *Let $(X_1, \ldots, X_m) \in \Omega^m$ and $(Y_1, \ldots, Y_m) \in \Omega^m$ denote independent random variables variables satisfying $d_{TV}(X_i, Y_i) \leq \epsilon$ for $i \in [m]$. Then*

$$d_{TV}((X_1, \ldots, X_m), (Y_1, \ldots, Y_m)) \leq m\epsilon$$

For two random variables $X, Y \in \Omega$, we write $X \equiv_D Y$ if X and Y are identically distributed (i.e. have the same probability distribution). For a continuous random variable X, we let $\mathrm{d}\Pr[X = x]$, for $x \in \Omega$, denote its *probability density* at x.

Algorithms on Convex Bodies: For the purposes of our sieving algorithm, we will need an algorithm to sample uniform points from K. We call a random vector $X \in K$ *η-uniform* if the total variation distance between X and a uniform vector on K is at most η. The following result of [26] provides the result:

Theorem 6 (Uniform Sampler) *Given $\eta > 0$, there exists an algorithm which outputs an η-*uniform *$X \in K$ using at most $\mathrm{poly}(n, \ln(\frac{1}{\eta}), \ln(\frac{R}{r}))$ calls to the oracle and arithmetic operations.*

Our main IP algorithm will provide a guarantee with respect to the barycenter of K. The following standard lemma shows that one can approximate a point near $\mathbf{b}(K)$ with overwhelming probability via sampling methods. We include a proof of this in the appendix for completeness.

Lemma 2 (Approx. Barycenter) *For $\epsilon > 0$, let $b = \frac{1}{N}\sum_{i=1}^{N} X_i$, $N = \left(\frac{2c}{\epsilon}\right)^2 n$, $c > 0$ an absolute constant, and where $X_1, \ldots, X_N$ are iid 4^{-n}-uniform samples on $K \subseteq \mathbb{R}^n$. Then*

$$\Pr[\| \pm (\mathbf{b} - \mathbf{b}(K))\|_{K-\mathbf{b}(K)} > \epsilon] \leq 2^{-n}$$

Lattices: An n-dimensional lattice $\mathcal{L} \subseteq \mathbb{R}^n$ is formed by integral combinations of linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^n$. Letting $B = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$, for a point $\mathbf{x} \in \mathbb{R}^n$ we define the *modulus operator* as

$$\mathbf{x} \pmod B = B(B^{-1}\mathbf{x} - \lfloor B^{-1}\mathbf{x} \rfloor)$$

where for $\mathbf{y} \in \mathbb{R}^n$, $\lfloor \mathbf{y} \rfloor = (\lfloor \mathbf{y}_1 \rfloor, \ldots, \lfloor \mathbf{y}_n \rfloor)$. We note that $\mathbf{x} \pmod B \in B[0,1)^n$, i.e. the *fundamental parallelipiped* of B and that $\mathbf{x} - (\mathbf{x} \pmod B) \in \mathcal{L}$, hence $\mathbf{x} \pmod B$ is the unique representative of the coset $\mathbf{x} + \mathcal{L}$ in $B[0,1)^n$.

Convex Geometry:

Lemma 3 *Take* $\mathbf{x}, \mathbf{y} \in K$ *satisfying* $\| \pm (\mathbf{x} - \mathbf{y})\|_{K-\mathbf{y}} \leq \alpha < 1$. *Then for* $\mathbf{z} \in \mathbb{R}^n$ *we have that*

1. *for* $\tau \geq 0$, $\mathbf{z} \in \tau K + (1-\tau)\mathbf{y} \Leftrightarrow \|\mathbf{z} - \mathbf{y}\|_{K-\mathbf{y}} \leq \tau$
2. $\|\mathbf{z} - \mathbf{y}\|_{K-\mathbf{y}} \leq \|\mathbf{y} - \mathbf{x}\|_{K-\mathbf{x}} + \alpha \ |1 - \|\mathbf{z} - \mathbf{x}\|_{K-\mathbf{x}}|$
3. $\|\mathbf{z} - \mathbf{x}\|_{K-\mathbf{x}} \leq \|\mathbf{z} - \mathbf{y}\|_{K-\mathbf{y}} + \frac{\alpha}{1-\alpha} \ |1 - \|\mathbf{z} - \mathbf{y}\|_{K-\mathbf{y}}|$

The following theorem of Milman and Pajor, tells us that $K - \mathbf{b}(K)$ is $\frac{1}{2}$-symmetric.

Theorem 7 ([27]) *Assume* $\mathbf{b}(K) = 0$. *Then* $\mathrm{vol}_n(K \cap -K) \geq \frac{1}{2^n}\mathrm{vol}_n(K)$.

Using the above theorem, we give a simple extension which shows that near-symmetry is a stable property.

Corollary 1 *Assume* $\mathbf{b}(K) = 0$. *Then for* $\mathbf{x} \in K$ *we have that* $K - \mathbf{x}$ *is* $\frac{1}{2}(1 - \|\mathbf{x}\|_K)$*-symmetric.*

Proofs of Lemma 3 and Corollary 1 are included in the appendix.

3 Algorithms

3.1 Integer Programming

For the first result in this section, we give a reduction from Approximate Integer Programming to the Approximate Closest Vector Problem.

Proof (Theorem 1: Approximate Integer Programming) We are given $0 < \epsilon \leq \frac{1}{2}$, and we wish to find a lattice point in $((1+\epsilon)K - \epsilon\mathbf{b}(K)) \cap \mathcal{L}$ or decide that $K \cap \mathcal{L} = \emptyset$. The algorithm, which we denote by $\mathrm{ApproxIP}(K, \mathcal{L}, \epsilon)$, will be the following:

Algorithm:

1. Compute $\mathbf{b} \in K$, satisfying $\| \pm (\mathbf{b} - \mathbf{b}(K))\|_{K-\mathbf{b}(K)} \leq \frac{1}{3}$, using Lemma 2.
2. Compute $\mathbf{y} \in \mathcal{L}$ such that $\mathbf{y}$ is $1 + \frac{2\epsilon}{5}$ approximate closest lattice vector to $\mathbf{b}$ under the norm $\|\cdot\|_{K-\mathbf{b}}$ using Approx-CVP (Theorem 9).
3. Return $\mathbf{y}$ if $\mathbf{y} \in \|\mathbf{y} - \mathbf{b}\|_{K-\mathbf{b}} \leq 1 + \frac{3\epsilon}{4}$, and otherwise return EMPTY (i.e. $K \cap \mathcal{L} = \emptyset$).

Correctness: Assuming that steps (1) and (2) return correct outputs (which occurs with overwhelming probability), we show that the final output is correct.

First note that if $\|\mathbf{y} - \mathbf{b}\|_{K-\mathbf{b}} \leq 1 + \frac{3\epsilon}{4}$, then by Lemma 3 we have that

$$\begin{aligned}\|\mathbf{y} - \mathbf{b}(K)\|_{K-\mathbf{b}(K)} &\leq \|\mathbf{y} - \mathbf{b}\|_{K-\mathbf{b}} + \frac{1}{3}\,|1 - \|\mathbf{y} - \mathbf{b}\|_{K-\mathbf{b}}| \\ &\leq 1 + \frac{3\epsilon}{4} + \frac{1}{3}\ \frac{3\epsilon}{4} = 1 + \epsilon\end{aligned}$$

as required. Now assume that $K \cap \mathcal{L} \neq \emptyset$. Then we can take $\mathbf{z} \in \mathcal{L}$ such that $\|\mathbf{z} - \mathbf{b}\|_{K-\mathbf{b}} \leq 1$. Since $\mathbf{y}$ is a $1 + \frac{2\epsilon}{5}$ closest vector, we must have that $\|\mathbf{y} - \mathbf{b}\|_{K-b} \leq 1 + \frac{2\epsilon}{5}$. Hence by the reasoning in the previous paragraph, we have that $\|\mathbf{y} - \mathbf{b}(K)\|_{K-\mathbf{b}(K)} \leq 1 + \epsilon$ as needed.

For the furthermore, we assume that $\frac{1}{1+\epsilon}K + \frac{\epsilon}{1+\epsilon}\mathbf{b}(K) \cap \mathcal{L} \neq \emptyset$. So we may pick $\mathbf{z} \in \mathcal{L}$ such that $\|\mathbf{z} - \mathbf{b}(K)\|_{K-\mathbf{b}(K)} \leq \frac{1}{1+\epsilon}$. By Lemma 3, we have that

$$\begin{aligned}\|\mathbf{z} - \mathbf{b}\|_{K-\mathbf{b}} &\leq \|\mathbf{z} - \mathbf{b}(K)\|_{K-\mathbf{b}(K)} + \frac{\frac{1}{3}}{1 - \frac{1}{3}} \left|1 - \|\mathbf{z} - \mathbf{b}(K)\|_{K-\mathbf{b}(K)}\right| \\ &\leq \frac{1}{1+\epsilon} + \frac{1}{2}\,\frac{\epsilon}{1+\epsilon} = \frac{1 + \frac{\epsilon}{2}}{1+\epsilon}\end{aligned}$$

Next by the assumptions on $\mathbf{y}$, we have that $\|\mathbf{y} - \mathbf{b}\|_{K-\mathbf{b}} \leq \frac{1+\frac{\epsilon}{2}}{1+\epsilon}(1 + \frac{2\epsilon}{5}) \leq 1$ since $0 < \epsilon \leq \frac{1}{2}$. Hence $\mathbf{y} \in K \cap \mathcal{L}$ as needed.

Runtime: For step (1), by Lemma 2 we can compute $\mathbf{b} \in K$, satisfying $\| \pm (\mathbf{b} - \mathbf{b}(K))\|_{K-\mathbf{b}(K)} \leq \frac{1}{3}$, with probability at least $1 - 2^{-n}$, by letting $\mathbf{b}$ be the average of $O(n)\ \ 4^{-n}$-uniform samples over K. By Theorem 6, each of these samples can be computed in $\text{poly}(n, \ln(\frac{R}{r}))$ time.

For step (2), we first note that by Corollary 1, $K - \mathbf{b}$ is $(1 - \frac{1}{3})\frac{1}{2} = \frac{1}{3}$-symmetric. Therefore, the call to the Approximate CVP algorithm, with error parameter $\frac{2\epsilon}{5}$ returns a valid approximation vector with probability at least $1 - 2^{-n}$ in time $O(3(\frac{5}{2\epsilon})^2)^n = 2^{O(n)}(1/\epsilon^2)^n$. Hence the entire algorithm takes time $2^{O(n)}(1/\epsilon^2)^n$ and outputs a correct answer with probability at least $1 - 2^{-n+1}$ as needed.

We now provide the reduction from Approximate Integer Optimization to Approximate Integer Programming Feasibility. To achieve this we show that Algorithm 1 (ApproxOPT), which proceeds by an essentially standard binary search, satisfies the desired specifications for Theorem 2.

To analyze ApproxOPT, we will require the following technical lemma. Due to the tedious nature of the proof, we delay its presentation till the appendix.

Lemma 4 (Well-Centered Bodies) *Let $K_{a,b} = K \cap \{\mathbf{x} \in \mathbb{R}^n : a \leq \langle \mathbf{x}, \mathbf{v} \rangle \leq b\}$, where K is as in Algorithm 1. Then during the execution of ApproxOpt, every call of ApproxIP is executed on a $(\mathbf{a}'_0, r', R')$-centered convex body $K_{a,b}$, where $r' \geq \frac{3r\delta}{64R\|\mathbf{v}\|_2}$, $R' \leq 2R$, and $\mathbf{a}'_0$ can be computed in polynomial time as a convex combination of $\mathbf{a}_0, \mathbf{x}_u$, and $\mathbf{x}_l$.*

Proof (Theorem 2: Approximate Integer Optimization) We are given $\mathbf{v} \in \mathbb{R}^n$, $0 < \epsilon \leq \frac{1}{2}$, and $\delta > 0$ where we wish to find a lattice point in $K + \epsilon(K - K) \cap \mathcal{L}$ whose objective value is within an additive δ of the best point in $K \cap \mathcal{L}$. We remember that K is $(\mathbf{a}_0, r, R)$-centered. Since we lose nothing by making δ smaller, we shall assume that $\delta \leq \frac{1}{64}\|\mathbf{v}\|_2 r$. This will allow us to assume that $\langle \mathbf{v}, \mathbf{x}_u \rangle - \langle \mathbf{v}, \mathbf{x}_l \rangle > 127\delta$ (see Lemma 4), i.e. there is significant space between the upper and lower bound for the continuous relaxation. We will show that Algorithm 1 correctly solves the optimization problem.

Algorithm 1 Algorithm ApproxOPT$(K, \mathcal{L}, \mathbf{v}, \epsilon, \delta)$

Input: $(\mathbf{a}_0, r, R)$-centered convex body $K \subseteq \mathbb{R}^n$ presented by membership oracle, lattice $\mathcal{L} \subseteq \mathbb{R}^n$ given by a basis, objective $\mathbf{v} \in \mathbb{R}^n$, tolerance parameters $0 < \epsilon \leq \frac{1}{2}$ and $0 < \delta \leq \frac{r\|\mathbf{v}\|_2}{64}$.

Output: EMPTY if $K \cap \mathcal{L} = \emptyset$ or $\mathbf{z} \in K + \epsilon(K - K) \cap \mathcal{L}$ satisfying $\sup_{\mathbf{y} \in K \cap \mathcal{L}} \langle \mathbf{v}, \mathbf{y} \rangle \leq \langle \mathbf{v}, \mathbf{z} \rangle + \delta$

1: $\mathbf{z} \leftarrow$ApproxIP$(K, \mathcal{L}, \epsilon)$
2: **if** $\mathbf{z} =$ EMPTY **then**
3: **return** EMPTY
4: Compute $\mathbf{x}_l, \mathbf{x}_u \in K$ using the ellipsoid algorithm satisfying $\inf_{\mathbf{x} \in K} \langle \mathbf{v}, \mathbf{x} \rangle \geq \langle \mathbf{v}, \mathbf{x}_l \rangle - \frac{\delta}{16}$ and $\sup_{\mathbf{x} \in K} \langle \mathbf{v}, \mathbf{x} \rangle \leq \langle \mathbf{v}, \mathbf{x}_u \rangle + \frac{\delta}{16}$
5: Set $l \leftarrow \langle \mathbf{v}, \mathbf{z} \rangle$ and $u \leftarrow \langle \mathbf{v}, \mathbf{x}_u \rangle + \frac{\delta}{16}$
6: **while** $u - l > \delta$ **do**
7: $m \leftarrow \frac{1}{2}(u + l)$
8: $\mathbf{y} \leftarrow$ApproxIP$(K \cap \{\mathbf{x} \in \mathbb{R}^n : m \leq \langle \mathbf{v}, \mathbf{x} \rangle \leq u\}, \mathcal{L}, \epsilon)$
9: **if** $\mathbf{y} =$ EMPTY **then**
10: $u \leftarrow m$
11: **if** $u < \langle \mathbf{x}_l, \mathbf{v} \rangle - \frac{\delta}{16}$ **return** $\mathbf{z}$; **else** $u \leftarrow \max\{u, \langle \mathbf{x}_l, \mathbf{v} \rangle + \frac{3\delta}{16}\}$
12: **else**
13: Set $\mathbf{z} \leftarrow \mathbf{y}$ and $l \leftarrow \langle \mathbf{v}, \mathbf{z} \rangle$
14: **return** $\mathbf{z}$

Correctness: Assuming that all the calls to the ApproxIP solver output a correct result (which occurs with overwhelming probability), we show that Algorithm 1 is correct. As can be seen, the algorithm performs a standard binary search over the objective value. During the iteration of the while loop, the value u represents the current best upper bound on $\sup_{\mathbf{y} \in K \cap \mathcal{L}} \langle \mathbf{v}, \mathbf{y} \rangle$, where this bound is achieved first by bounding $\sup_{\mathbf{x} \in K} \langle \mathbf{v}, \mathbf{x} \rangle$ (line 5), or by showing the lattice infeasibility of appriopriate restrictions of K (line 8). On line 11, the value u can potentially be relaxed to $\max\{u, \langle \mathbf{x}_l, \mathbf{v} \rangle + \frac{3\delta}{16}\}$, though this operation maintains that u is an upper bound. We remark that the value of u is relaxed here to guarantee that the restrictions of K on which ApproxIP is called are all well-centered (the details of this are presented in Lemma 4). Similarly, the value l represents the objective value of the best lattice point found thus far, which is denoted by $\mathbf{z}$. Now as long as the value of $\mathbf{z}$ is not null, we claim that $\mathbf{z} \in K + \epsilon(K - K)$. To see this note that $\mathbf{z}$ is the output of some call to Approx IP, on $K_{a,b} = K \cap \{\mathbf{x} \in \mathbb{R}^n : a \leq \langle \mathbf{v}, \mathbf{x} \rangle \leq b\}$ for some $a < b$, the lattice $\mathcal{L}$, with tolerance parameter ϵ. Hence if $\mathbf{z}$ is not null, we are guaranteed that

$$\begin{aligned} \mathbf{z} \in (1+\epsilon)K_{a,b} - \epsilon\mathbf{b}(K_{a,b}) &= K_{a,b} + \epsilon(K_{a,b} - \mathbf{b}(K_{a,b})) \\ &\subseteq K_{a,b} + \epsilon(K_{a,b} - K_{a,b}) \subseteq K + \epsilon(K - K) \end{aligned} \tag{3}$$

since $\mathbf{b}(K_{a,b}) \in K_{a,b} \subseteq K$. Therefore $\mathbf{z} \in K + \epsilon(K - K)$ as required. Now, the algorithm returns EMPTY if $K \cap \mathcal{L} = \emptyset$ (line 3), or $\mathbf{z}$ if $u < \langle \mathbf{x}_l, \mathbf{v} \rangle - \frac{\delta}{16}$ or if $u - l < \delta$ (line 14). Note that if $u < \langle \mathbf{x}_l, \mathbf{v} \rangle - \frac{\delta}{16}$, then since u is a valid upper bound on $\max_{\mathbf{w} \in K \cap \mathcal{L}} \langle \mathbf{w}, \mathbf{v} \rangle$ and since

$$u < \langle \mathbf{x}_l, \mathbf{v} \rangle - \frac{\delta}{16} \leq \min_{\mathbf{y} \in K} \langle \mathbf{y}, \mathbf{v} \rangle$$

we must have that $K \cap \mathcal{L} = \emptyset$. Therefore the returned vector $\mathbf{z}$ clearly has higher objective value than any vector in $K \cap \mathcal{L} = \emptyset$. Lastly, if $u - l < \delta$, the returned vector $\mathbf{z} \in K + \epsilon(K - K)$ is indeed nearly optimal. The algorithm's output is therefore valid as required.

Runtime: Assuming that each call to ApproxIP returns a correct result, we shall bound the number of iterations of the while loop. After this, using a union bound over the failure probability of ApproxIP, we get a bound on the probability that the algorithm does not perform as described by the analysis.

First, we show that gap $u - l$ decreases by a factor of at least $\frac{3}{4}$ after each non-exiting iteration of the loop. We first examine the case where $K_{m,u}$ is declared EMPTY in line 8. Let u_0 denote the value of u on line 7. After line 10, note that $u = m = \frac{1}{2}(l + u_0)$. Since the loop exits if $u < \langle \mathbf{x}_l, \mathbf{v} \rangle - \frac{\delta}{16}$ on line 11, we may assume that this is not the case. If $\langle \mathbf{x}_l, \mathbf{v} \rangle + \frac{3\delta}{16} \leq u$, note that u is unchanged after line 11. Therefore $u - l = \frac{1}{2}(u_0 + l) - l = \frac{1}{2}(u_0 - l)$ decreases by a factor $\frac{1}{2}$. Otherwise, we have that $\langle \mathbf{x}_l, \mathbf{v} \rangle - \frac{\delta}{16} \leq \frac{1}{2}(u_0 + l) < \langle \mathbf{x}_l, \mathbf{v} \rangle + \frac{3\delta}{16}$, and the value of u after line 11 is set to $\langle \mathbf{x}_l, \mathbf{v} \rangle + \frac{3\delta}{16}$. From the while loop invariant, note that $u_0 - l \geq \delta$. Therefore, after line 11, we have that

$$\begin{aligned} u - l = \langle \mathbf{x}_l, \mathbf{v} \rangle + \frac{3\delta}{16} - l &= \left(\langle \mathbf{x}_l, \mathbf{v} \rangle + \frac{3\delta}{16} - \frac{1}{2}(u_0 + l) \right) + \frac{1}{2}(u_0 - l) \\ &\leq \frac{1}{4}\delta + \frac{1}{2}(u_0 - l) \leq \frac{3}{4}(u_0 - l) \end{aligned}$$

as needed. Next, if a lattice point y is returned in line 8, we know by equation (3) that $\mathbf{y} \in K_{m,u} - \epsilon(K_{m,u} - K_{m,u})$. Therefore

$$\langle \mathbf{v}, \mathbf{y} \rangle \geq \inf_{\mathbf{x} \in K_{m,u}} \langle \mathbf{v}, \mathbf{y} \rangle - \epsilon \left(\sup_{\mathbf{x} \in K_{m,u}} \langle \mathbf{v}, \mathbf{y} \rangle - \inf_{\mathbf{x} \in K_{m,u}} \langle \mathbf{v}, \mathbf{y} \rangle \right) \geq m - \epsilon(u - m) \quad (4)$$

Since $m = \frac{1}{2}(l + u)$, and $\epsilon \leq \frac{1}{2}$, we see that

$$u - \langle \mathbf{v}, \mathbf{y} \rangle \leq (u - m) + \epsilon(u - m) \leq \frac{1}{2}(u - l) + \frac{1}{4}(u - l) = \frac{3}{4}(u - l)$$

as needed. From here, we claim that we perform at most $\lceil \ln(\frac{4R\|\mathbf{v}\|_2)}{\ln(\delta)} / \ln(\frac{4}{3}) \rceil$ iterations of the for loop. Now since $K \subseteq \mathbf{a}_0 + RB_2^n$, note that $\text{width}_K(\mathbf{v}) \leq 2R\|\mathbf{v}\|_2$. Therefore, using equation (4), the initial value of $u - l$ (line 6) is at most

$$2R\|\mathbf{v}\|_2 + \epsilon(2R\|\mathbf{v}\|_2) + \frac{\delta}{16} \leq 2R\|\mathbf{v}\|_2 + \frac{1}{2}(2R\|\mathbf{v}\|_2) + \frac{2r\|\mathbf{v}\|}{16} \leq 4R\|\mathbf{v}\|_2$$

Since $u - l$ decreases by a factor at least $\frac{3}{4}$ at each iteration, it takes at most $\lceil \ln(\frac{4R\|\mathbf{v}\|_2}{\delta}) / \ln \frac{4}{3} \rceil$ iterations before $u - l \leq \delta$. Since we call ApproxIP at most twice at each iteration, the probability that any one of these calls fails (whereupon the above analysis does not hold) is at most $2\lceil \ln(\frac{4R\|\mathbf{v}\|_2}{\delta}) / \ln \frac{4}{3} \rceil F$,

where F is the failure probability of a call to ApproxIP. For the purposes of this algorithm, we claim that the error probability of a call to ApproxIP can be made arbitrarily small by repetition. To see this, note any lattice vector returned by ApproxIP$(K_{a,b}, \mathcal{L}, \epsilon)$ is always a success for our purposes, since by the algorithm's design any returned vector is always in $K_{a,b} + \epsilon(K_{a,b} - K_{a,b}) \cap \mathcal{L}$ (which is sufficient for us). Hence the only failure possibility is that ApproxIP returns that $K_{a,b} \cap \mathcal{L} = \emptyset$ when this is not the case. By the guarantees on ApproxIP, the probability that this occurs over k independent repetitions is at most 2^{-nk}. Hence by repeating each call to ApproxIP $O(1 + \frac{1}{n} \ln \ln \frac{R\|\mathbf{v}\|}{\delta})$ times, the total error probability over all calls can be reduced to 2^{-n} as required. Hence with probability at least $1 - 2^{-n}$, the algorithm correctly terminates in at most $\lceil \ln(\frac{4R\|\mathbf{v}\|_2}{\delta}) / \ln \frac{4}{3} \rceil$ iterations of the while loop.

Lastly, we must check that each call to ApproxIP is done over a well-centered body $K_{a,b}$, i.e. we must be able to provide to ApproxIP a center $\mathbf{a}'_0 \in \mathbb{R}^n$ and radii r', R' such that $\mathbf{a}'_0 + r'B_2^n \subseteq K_{a,b} \subseteq \mathbf{a}'_0 + R'B_2^n$, where each of $\mathbf{a}'_0, r', R'$ have size polynomial in the input parameters. This is guaranteed by Lemma 4.

Given the above, since we call ApproxIP at most once at each iteration (over a well-centered convex body), with probability at least $1 - 2^{-n}$ the total running time is $2^{O(n)}(\frac{1}{\epsilon^2})^n \operatorname{polylog}(R, r, \delta, \|\mathbf{v}\|_2)$ and the space usage is $2^{O(n)}(\frac{1}{\epsilon})^n$ as required.

For the last result of this section, we give our implementation of the Flatness theorem.

Proof (Theorem 4: Algorithmic Flatness Theorem) Take ϵ, $0 \leq \epsilon \leq \frac{1}{2}$, $K \subseteq \mathbb{R}^n$ a convex body, and $\mathcal{L}$ an n-dimensional lattice. Here we wish to either find a lattice point in K, decide that K is lattice free, or return a small lattice width direction for K. We shall achieve this with the following algorithm:

Algorithm:

1. Compute $\mathbf{y} \leftarrow$ ApproxIP$(K, \mathcal{L}, \epsilon)$.
2. If $\mathbf{y} =$ EMPTY, return EMPTY. Else if $\mathbf{y} \in K \cap \mathcal{L}$, return $\mathbf{y}$.
3. Else, use a symmetric norm SVP solver to compute $\mathbf{y} \in \mathcal{L}^* \setminus \{\mathbf{0}\}$ such that $\operatorname{width}_K(\mathbf{y}) = \operatorname{width}(K, \mathcal{L})$. Return $\mathbf{y}$.

Correctness: We will prove correctness of the algorithm assuming that our call to ApproxIP and to the SVP solver return correct outputs (which occurs with overwhelming probability). First, note that if $((1+\epsilon)K - \epsilon\mathbf{b}(K)) \cap \mathcal{L} = \emptyset$, by the guarantees on ApproxIP, we must have that ApproxIP declares $K \cap \mathcal{L} = \emptyset$. Furthermore, if $\left(\frac{1}{1+\epsilon}K - \frac{\epsilon}{1+\epsilon}\mathbf{b}(K)\right) \cap \mathcal{L} \neq \emptyset$, then again by the guarantees on ApproxIP, we have that ApproxIP returns $\mathbf{y} \in K \cap \mathcal{L}$. This proves that the first two guarantees of Theorem 4 are satisfied.

Let us now assume that both of the preliminary tests fail, and that the algorithm computes a shortest non-zero vector $\mathbf{y}$ in $\mathcal{L}^*$ under the norm $\operatorname{width}_K(\cdot)$. Here, we must show that $\operatorname{width}_K(\mathbf{y}) = \operatorname{width}(K, \mathcal{L}) \leq (1+\epsilon)\operatorname{Fl}(K)$.

To see this, note that by the guarantees on ApproxIP, we must have that $(\frac{1}{1+\epsilon}K - \frac{\epsilon}{1+\epsilon}\mathbf{b}(K)) \cap \mathcal{L} = \emptyset$. Therefore, we see that

$$\mathrm{Fl}(K) \geq \mathrm{width}\left(\frac{1}{1+\epsilon}K - \frac{\epsilon}{1+\epsilon}\mathbf{b}(K), \mathcal{L}\right) = \frac{1}{1+\epsilon}\mathrm{width}(K, \mathcal{L})$$

as needed.

Runtime: The algorithm makes one call to ApproxIP and one call to a symmetric norm SVP solver. The call to ApproxIP requires $2^{O(n)}(\frac{1}{\epsilon^2})^n$-time and $2^{O(n)}(\frac{1}{\epsilon})^n$-space, and successfully returns a correct solution with probability at least $1 - 2^{-n}$. Next, making a call to the AKS based symmetric SVP solver of Arvind and Joglekar [14] requires at most $2^{O(n)}$-time and space, and returns a correct solution probability at least $1 - 2^{-n}$. We note that since K is well-centered, the norm $\mathrm{width}_K(\mathbf{x})$, for $\mathbf{x} \in \mathbb{R}^n$, can be computed to within any desired accuracy in polynomial time using the ellipsoid algorithm. Therefore, the full algorithm requires at most $2^{O(n)}(\frac{1}{\epsilon^2})^n$-time and $2^{O(n)}(\frac{1}{\epsilon})^n$-space, and outputs a correct solution with probability at least $1 - 2^{-n+1}$, as needed.

3.2 Subspace Avoiding Problem

In the following two sections, $C \subseteq \mathbb{R}^n$ will denote be a $(\mathbf{0}, r, R)$-centered γ-symmetric convex body, and $\mathcal{L} \subseteq \mathbb{R}^n$ will denote an n-dimensional lattice.

In this section, we introduce the Subspace Avoiding Problem of [13], and outline how the AKS sieve can be adapted to solve it under general norms.

Let $M \subseteq \mathbb{R}^n$ be a linear subspace where $\dim(M) = k \leq n-1$. Let $\lambda(C, \mathcal{L}, M) = \inf_{\mathbf{x} \in \mathcal{L} \setminus M} \|\mathbf{x}\|_C$. Note that under this definition, we have the identity $\lambda_1(C, \mathcal{L}) = \lambda(C, \mathcal{L}, \{0\})$.

Definition 1 The $(1+\epsilon)$-Approximate Subspace Avoiding Problem (SAP) with respect C, $\mathcal{L}$ and M is to find a lattice vector $\mathbf{y} \in \mathcal{L} \setminus M$ such that $\|\mathbf{y}\|_C = (1+\epsilon)\lambda(C, \mathcal{L}, M)$.

For $\mathbf{x} \in \mathbb{R}^n$, let $\|\mathbf{x}\|_C^* = \min\{\|\mathbf{x}\|_C, \|\mathbf{x}\|_{-C}\}$. For a point $\mathbf{x} \in \mathbb{R}^n$, define $s(\mathbf{x}) = 1$ if $\|\mathbf{x}\|_C \leq \|\mathbf{x}\|_{-C}$ and $s(\mathbf{x}) = -1$ if $\|\mathbf{x}\|_C > \|\mathbf{x}\|_{-C}$. From the notation, we have that $\|\mathbf{x}\|_C^* = \|\mathbf{x}\|_{s(\mathbf{x})C} = \|s(\mathbf{x})\mathbf{x}\|_C$.

We begin with an extension of the AKS sieving lemma to the asymmetric setting. The following lemma will provide the central tool for the SAP algorithm.

Lemma 5 (Basic Sieve) *Let $(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_N, \mathbf{y}_N) \in \mathbb{R}^n \times \mathbb{R}^n$ denote a list of pairs satisfying $\mathbf{y}_i - \mathbf{x}_i \in \mathcal{L}$, $\|\mathbf{x}_i\|_C^* \leq \beta$ and $\|\mathbf{y}_i\|_C^* \leq D \ \ \forall i \in [N]$. Then a clustering, $c : \{1, \ldots, N\} \to J$, $J \subseteq [N]$, satisfying:*

1. $|J| \leq 2\left(\frac{5}{\gamma}\right)^n$
2. $\|\mathbf{y}_i - \mathbf{y}_{c(i)} + \mathbf{x}_{c(i)}\|_C^* \leq \frac{1}{2}D + \beta$
3. $\mathbf{y}_i - \mathbf{y}_{c(i)} + \mathbf{x}_{c(i)} - \mathbf{x}_i \in \mathcal{L}$

for all $i \in [N] \setminus J$, can be computed in deterministic $O(N\left(\frac{5}{\gamma}\right)^n)$-time.

Proof

Algorithm: We build the set J and clustering c iteratively, starting from $J = \emptyset$, in the following manner. For each $i \in [N]$, check if there exists $j \in J$ such that $\|\mathbf{y}_i - \mathbf{y}_j\|_{s(\mathbf{x}_j)C} \leq \frac{D}{2}$. If such a j exists, set $c(i) = j$. Otherwise, append i to the set J and set $c(i) = i$. Repeat.

Analysis: We first note, that for any $i, j \in [N]$, we have that $\mathbf{y}_i - \mathbf{y}_j + \mathbf{x}_j - \mathbf{x}_i = (\mathbf{y}_i - \mathbf{x}_i) - (\mathbf{y}_j - \mathbf{x}_j) \in \mathcal{L}$ since by assumption both $\mathbf{y}_i - \mathbf{x}_i, \mathbf{y}_j - \mathbf{x}_j \in \mathcal{L}$. Hence, property (3) is trivially satisfied by the clustering c.

We now check that the clustering satisfies property (2). For $i \in [N] \setminus J$, note that by construction we have that $\|\mathbf{y}_i - \mathbf{y}_{c(i)}\|_{sC} \leq \frac{D}{2}$ where $s = s(\mathbf{x}_{c(i)})$. Therefore by the triangle inequality, we have that

$$\begin{aligned}\|\mathbf{y}_i - \mathbf{y}_{c(i)} + \mathbf{x}_{c(i)}\|_C^* &\leq \|\mathbf{y}_i - \mathbf{y}_{c(i)} + \mathbf{x}_{c(i)}\|_{sC} \leq \|\mathbf{y}_i - \mathbf{y}_{c(i)}\|_{sC} + \|\mathbf{x}_{c(i)}\|_{sC} \\ &= \|\mathbf{y}_i - \mathbf{y}_{c(i)}\|_{sC} + \|\mathbf{x}_{c(i)}\|_C^* \leq \frac{D}{2} + \beta\end{aligned}$$

as required.

We now show that J satisfies property (1). By construction of J, we know that for $i, j \in J$, $i < j$ that $\|\mathbf{y}_j - \mathbf{y}_i\|_{s(\mathbf{x}_i)C} > \frac{D}{2}$. Therefore we have that

$$\|\mathbf{y}_j - \mathbf{y}_i\|_{s(\mathbf{x}_i)C} > \frac{D}{2} \Rightarrow \|\mathbf{y}_j - \mathbf{y}_i\|_{C \cap -C} = \|\mathbf{y}_i - \mathbf{y}_j\|_{C \cap -C} > \frac{D}{2},$$

where the last equality follows by symmetry of $C \cap -C$. We now claim that

$$\mathbf{y}_i + \frac{D}{4}(C \cap -C) \cap \mathbf{y}_j + \frac{D}{4}(C \cap -C) = \emptyset. \tag{5}$$

Assume not, then we may pick $\mathbf{z}$ in the intersection above. Then by assumption

$$\begin{aligned}\|\mathbf{y}_j - \mathbf{y}_i\|_{C \cap -C} &= \|(\mathbf{y}_j - \mathbf{z}) + (\mathbf{z} - \mathbf{y}_i)\|_{C \cap -C} \leq \|\mathbf{y}_j - \mathbf{z}\|_{C \cap -C} + \|\mathbf{z} - \mathbf{y}_i\|_{C \cap -C} \\ &= \|\mathbf{z} - \mathbf{y}_j\|_{C \cap -C} + \|\mathbf{z} - \mathbf{y}_i\|_{C \cap -C} \leq \frac{D}{4} + \frac{D}{4} = \frac{D}{2}\end{aligned}$$

a clear contradiction.

For each $i \in [N]$, by assumption $\|\mathbf{y}_i\|_C^* \leq D \Leftrightarrow \mathbf{y}_i \in D(C \cup -C)$. Therefore, we get that

$$\begin{aligned}\mathbf{y}_i + \frac{D}{4}(C \cap -C) &\subseteq D(C \cup -C) + \frac{D}{4}(C \cap -C) \\ &= D((C + \frac{1}{4}(C \cap -C)) \cup (-C + \frac{1}{4}(C \cap -C))) \\ &\subseteq D((C + \frac{1}{4}C) \cup (-C + \frac{1}{4}(-C))) = \frac{5}{4}D(C \cup -C)\end{aligned} \tag{6}$$

From (5), (6), and since $J \subseteq [N]$, we have that

$$|J| = \frac{\mathrm{vol}_n(\{\mathbf{y}_i : i \in J\} + \frac{D}{4}(C \cap -C))}{\mathrm{vol}_n(\frac{D}{4}(C \cap -C))} \leq \frac{\mathrm{vol}_n(\frac{5}{4}D(C \cup -C))}{\mathrm{vol}_n(\frac{D}{4}(C \cap -C))}$$
$$\leq \frac{\left(\frac{5}{4}\right)^n (\mathrm{vol}_n(DC) + \mathrm{vol}_n(-DC))}{\left(\frac{\gamma}{4}\right)^n \mathrm{vol}_n(DC)} = 2\left(\frac{5}{\gamma}\right)^n$$

as needed.

To bound the running time of the clustering algorithm is straightforward. For each element of $[N]$, we iterate once through the partially constructed set J. Since $|J| \leq 2\left(\frac{5}{\gamma}\right)^n$ throughout the entire algorithm, we have that the entire runtime is bounded by $O(N\left(\frac{5}{\gamma}\right)^n)$ as required.

Definition 2 (Sieving Procedure) For a list of pairs $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)$ as in Lemma 5, we call an application of the *Sieving Procedure* the process of computing the clustering $c : [N] \to J$, and outputting the list of pairs $(\mathbf{x}_i, \mathbf{y}_i - \mathbf{y}_{c(i)} + \mathbf{x}_{c(i)})$ for all $i \in [N] \setminus J$.

Note that the *Sieving Procedure* deletes the set of pairs associated with the cluster centers J, and combines the remaining pairs with their associated centers.

We remark some differences with the standard AKS sieve. Here the Sieving Procedure does not guarantee that $\|\mathbf{y}_i\|_C$ decreases after each iteration. Instead it shows that at least one of $\|\mathbf{y}_i\|_C$ or $\|-\mathbf{y}_i\|_C$ decreases appropriately at each step. Hence the region we must control is in fact $D(C \cup -C)$, which we note is generally non-convex. Additionally, our analysis shows that how well we can use $\|\cdot\|_C$ to sieve only depends on $\mathrm{vol}_n(C \cap -C)/\mathrm{vol}_n(C)$, which is a very flexible global quantity. For example, if $C = [-1, 1]^{n-1} \times [-1, 2^n]$ (i.e. a cube with one highly skewed coordinate) then C is still $\frac{1}{2}$-symmetric, and hence the sieve barely notices the asymmetry.

The algorithm for approximate SAP we describe presently will construct a list of large pairs as above, and use repeated applications of the *Sieving Procedure* to create shorter and shorter vectors.

We will need the following lemmas for the SAP algorithm.

Lemma 6 *Let $C \subseteq \mathbb{R}^n$ a $(\mathbf{0}, r, R)$-centered convex body, $\mathcal{L} \subseteq \mathbb{R}^n$ be an n-dimensional lattice, and $M \subseteq \mathbb{R}^n$, $\dim(M) \leq n-1$, be a linear subspace. Then a number $\nu > 0$ satisfying*

$$\nu \leq \lambda(C, \mathcal{L}, M) \leq 2^n \frac{R}{r} \nu$$

can be computed in polynomial time.

The above lemma follows directly from Lemma 4.1 of [13]. They prove it for ℓ_p balls, but it is easily adapted to the above setting using the relationship $\frac{1}{r}\|\mathbf{x}\|_2 \leq \|\mathbf{x}\|_C \leq \frac{1}{R}\|\mathbf{x}\|_2$ (since C is $(\mathbf{0}, r, R)$-centered).

Lemma 7 *Take* $\mathbf{v} \in \mathbb{R}^n$ *where* $\beta \leq \|\mathbf{v}\|_C \leq \frac{3}{2}\beta$. *Define* $C_{\mathbf{v}}^+ = \beta C \cap (\mathbf{v} - \beta C)$ *and* $C_{\mathbf{v}}^- = (\beta C - \mathbf{v}) \cap -\beta C$. *Then*

$$\frac{\mathrm{vol}_n(C_{\mathbf{v}}^+)}{\mathrm{vol}_n(\beta C)} = \frac{\mathrm{vol}_n(C_{\mathbf{v}}^-)}{\mathrm{vol}_n(\beta C)} \geq \left(\frac{\gamma}{4}\right)^n$$

Furthermore, $\mathrm{int}(C_{\mathbf{v}}^+) \cap \mathrm{int}(C_{\mathbf{v}}^-) = \emptyset$.

Proof Since $\|\mathbf{v}\|_C \leq \frac{3}{2}\beta$, we see that $\frac{\mathbf{v}}{2} \in \frac{3}{4}C$. Now we get that

$$\frac{\mathbf{v}}{2} + \frac{1}{4}\beta(C \cap -C) \subseteq \frac{3}{4}\beta C + \frac{1}{4}\beta C = \beta C$$

Furthermore since $\|\frac{\mathbf{v}}{2} - \mathbf{v}\|_{-C} = \|-\frac{\mathbf{v}}{2}\|_{-C} = \frac{1}{2}\|\mathbf{v}\|_C \leq \frac{3}{4}\beta$, we also have that $\frac{\mathbf{v}}{2} \in \mathbf{v} - \frac{3}{4}C$. Therefore

$$\frac{\mathbf{v}}{2} + \frac{1}{4}\beta(C \cap -C) \subseteq (\mathbf{v} - \frac{3}{4}\beta C) + \frac{1}{4}\beta(-C) = \mathbf{v} - \beta C$$

We therefore conclude that

$$\frac{\mathrm{vol}_n(\beta C \cap (\mathbf{v} - \beta C))}{\mathrm{vol}_n(\beta C)} \geq \frac{\mathrm{vol}_n(\frac{\mathbf{v}}{2} + \frac{1}{4}\beta(C \cap -C))}{\mathrm{vol}_n(\beta C)} = \frac{\mathrm{vol}_n(C \cap -C)}{4^n \mathrm{vol}_n(C)} \geq \left(\frac{\gamma}{4}\right)^n$$

as needed. For the furthermore, we remember that $\|\mathbf{x}\|_C = \inf\{s \geq 0 : \mathbf{x} \in sC\} = \sup\{\langle \mathbf{x}, \mathbf{y}\rangle : \mathbf{y} \in C^*\}$ and that $(-C)^* = -C^*$. Now assume there exists $\mathbf{x} \in C_{\mathbf{v}}^+ \cap C_{\mathbf{v}}^-$. Then $\mathbf{x} = \mathbf{v} - \beta\mathbf{k}_1 = \beta\mathbf{k}_2 - \mathbf{v}$ where $\mathbf{k}_1, \mathbf{k}_2 \in C$. Choose $\mathbf{y} \in C^*$ such that $\langle \mathbf{y}, \mathbf{v}\rangle = \|\mathbf{v}\|_C$. Note that

$$\begin{aligned}
\langle \mathbf{y}, \mathbf{v} - \beta\mathbf{k}_1 - (\beta\mathbf{k}_2 - \mathbf{v})\rangle &= 2\langle \mathbf{y}, \mathbf{v}\rangle - \beta(\langle \mathbf{y}, \mathbf{k}_1\rangle + \langle \mathbf{y}, \mathbf{k}_2\rangle) \\
&= 2\|\mathbf{v}\|_C - \beta(\langle \mathbf{y}, \mathbf{k}_1\rangle + \langle \mathbf{y}, \mathbf{k}_2\rangle) \\
&\geq 2\|\mathbf{v}\|_C - \beta(\|\mathbf{k}_1\|_C + \|\mathbf{k}_2\|_C) \geq 2\beta - 2\beta = 0
\end{aligned}$$

Since $\mathbf{v} - \beta\mathbf{k}_1 - (\beta\mathbf{k}_2 - \mathbf{v}) = \mathbf{x} - \mathbf{x} = 0$ by construction, all of the above inequalities must hold at equality. In particular, we must have that $1 = \|\mathbf{k}_1\|_C = \|\mathbf{k}_2\|_C = \langle \mathbf{y}, \mathbf{k}_1\rangle = \langle \mathbf{y}, \mathbf{k}_2\rangle$. Since $-\mathbf{y} \in (-C)^*$, we know that

$$\mathbf{v} - \beta C \subseteq \{\mathbf{x} \in \mathbb{R}^n : \langle -\mathbf{y}, \mathbf{x} - \mathbf{v}\rangle \leq \beta\}$$

and since $\langle -\mathbf{y}, (\mathbf{v} - \beta\mathbf{k}_1) - \mathbf{v}\rangle = \beta\langle \mathbf{y}, \mathbf{k}_1\rangle = \beta$, we must have that $\mathbf{v} - \beta\mathbf{k}_1 \in \partial C_{\mathbf{v}}^+$. Via a symmetric argument, we get that $\beta\mathbf{k}_2 - \mathbf{v} \in \partial C_{\mathbf{v}}^-$. Therefore $C_{\mathbf{v}}^+ \cap C_{\mathbf{v}}^- \subseteq \partial C_{\mathbf{v}}^+ \cap \partial C_{\mathbf{v}}^- \Leftrightarrow \mathrm{int}(C_{\mathbf{v}}^+) \cap \mathrm{int}(C_{\mathbf{v}}^-) = \emptyset$, as needed.

Algorithm ShortVectors above is the core subroutine for the SAP solver. We relate some important details about the SAP algorithm. Our algorithm for SAP follows a standard procedure. We first guess a value β satisfying $\beta \leq \lambda(C, \mathcal{L}, M) \leq \frac{3}{2}\beta$, and then run ShortVectors on inputs $C, \mathcal{L}, M, \beta$ and ϵ. We show that for this value of β, ShortVectors outputs a $(1+\epsilon)$ approximate solution with overwhelming probability.

As we can be seen above, the main task of the ShortVectors algorithm, is to generate a large quantity of random vectors, and sieve them until they

Algorithm 2 ShortVectors(C,$\mathcal{L}$,M,β,ϵ)

Input: $(\mathbf{0}, r, R)$-centered γ-symmetric convex body $C \subseteq \mathbb{R}^n$, basis $B = (\mathbf{b}_1, \ldots, \mathbf{b}_n) \in \mathbb{Q}^{n \times n}$ for $\mathcal{L}$, linear subspace $M \subseteq \mathbb{R}^n$, scaling parameter $\beta > 0$, tolerance parameter $0 < \epsilon \leq \frac{1}{2}$

1: $D \leftarrow n \max_{1 \leq i \leq n} \|\mathbf{b}_i\|_C$

2: $\mathbf{N}_0 \leftarrow 4\lceil 6 \ln\left(\frac{D}{\beta}\right)\rceil \left(\frac{20}{\gamma^2}\right)^n + 8\left(\frac{36}{\gamma^2 \epsilon}\right)^n$, $\eta \leftarrow \frac{2^{-(n+1)}}{\mathbf{N}_0}$

3: Create pairs (X_1^0, Y_1^0),(X_2^0, Y_2^0), $\ldots$,$(X_{\mathbf{N}_0}^0, Y_{\mathbf{N}_0}^0)$ as follows: for each $i \in [\mathbf{N}_0]$, compute Z an η-uniform sample over βC (using Theorem 6) and a uniform s in $\{-1, 1\}$, and set $X_i^0 \leftarrow sZ$ and $Y_i^0 \leftarrow X_i^0 \pmod B$.

4: $t \leftarrow 0$

5: **while** $D \geq 3\beta$ **do**

6: Apply *Sieving Procedure* to (X_1^t, Y_1^t), $\ldots$,$(X_{\mathbf{N}_t}^t, Y_{\mathbf{N}_t}^t)$ yielding (X_1^{t+1}, Y_1^{t+1}), $\ldots$,$(X_{\mathbf{N}_{t+1}}^{t+1}, Y_{\mathbf{N}_{t+1}}^{t+1})$

7: $D \leftarrow \frac{D}{2} + \beta$ and $t \leftarrow t + 1$

8: **return** $\{Y_i^t - X_i^t - (Y_j^t - X_j^t) : i, j \in [\mathbf{N}_t]\} \setminus M$

are all of relatively small size (i.e. $3\beta \leq 3\lambda(C, \mathcal{L}, M)$). ShortVectors then examines all the differences between the sieved vectors in the hopes of finding one of size $(1+\epsilon)\lambda(C, \mathcal{L}, M)$ in $\mathcal{L} \setminus M$. ShortVectors, in fact, needs to balance certain tradeoffs. On the one hand, it must sieve enough times to guarantee that the vector differences have small size. On the other, it must use "large" perturbations sampled from $\beta(C \cup -C)$, to guarantee that these differences do not all lie in M.

We note that the main algorithmic difference with respect to [13,14] is the use of a modified sieving procedure (i.e. that pays attention to asymmetry when building the clusters) as well as the use of a different sampling distribution for the perturbation vectors (i.e. over $\beta(C \cup -C)$ instead of just βC).

Theorem 8 (Approximate-SAP) *For $0 < \epsilon \leq \frac{1}{2}$, a lattice vector $\mathbf{y} \in \mathcal{L} \setminus M$ such that $\|\mathbf{y}\|_C \leq (1+\epsilon)\lambda(C, \mathcal{L}, M)$ can be computed in $2^{O(n)}(\frac{1}{\gamma^4 \epsilon^2})^n$-time and $2^{O(n)}(\frac{1}{\gamma^2 \epsilon})^n$-space with probability at least $1 - 2^{-n}$. Furthermore, if $\lambda(C, \mathcal{L}, M) \leq t\lambda_1(C, \mathcal{L})$, $t \geq 2$, a vector $\mathbf{y} \in \mathcal{L} \setminus M$ satisfying $\|\mathbf{y}\|_C = \lambda(C, \mathcal{L}, M)$, can be with computed in $2^{O(n)}\left(\frac{1}{\gamma^4 t^2}\right)^n$-time and $2^{O(n)}\left(\frac{1}{\gamma^2 t}\right)^n$-space with probability at least $1 - 2^{-n}$.*

Proof

Algorithm: The algorithm for $(1+\epsilon)$-SAP is as follows:

1. Using Lemma 6 compute a value ν satisfying $\nu \leq \lambda(C, \mathcal{L}, M) \leq 2^n \frac{R}{r}\nu$.
2. For each $i \in 0, 1, \ldots, \lceil \ln(2^n R/r)/\ln(3/2) \rceil$, let $\beta = (3/2)^i \nu$ and run ShortVectors$(C, \mathcal{L}, \beta, \epsilon)$.
3. Return the shortest vector found with respect to $\|\cdot\|_C$ in the above runs of ShortVectors.

Preliminary Analysis: In words, the algorithm first guesses a good approximation of $\lambda(C, \mathcal{L}, M)$ (among polynomially many choices) and runs the ShortVectors algorithm on these guesses. By design, there will be one iteration of the algorithm where β satisfies $\beta \leq \lambda(C, \mathcal{L}, M) \leq \frac{3}{2}\beta$. We prove that for this setting of β the algorithm returns a $(1+\epsilon)$-approximate solution to the SAP problem with probability at least $1 - 2^{-n}$.

Take $\mathbf{v} \in \mathcal{L} \setminus M$ denote an optimal solution to the SAP problem, i.e. $\mathbf{v}$ satisfies $\|\mathbf{v}\|_C = \lambda(C, \mathcal{L}, M)$. We will show that with probability at least $1-2^{-n}$, a small pertubation of $\mathbf{v}$ (Claim 4) will be in the set returned by ShortVectors when run on inputs C,$\mathcal{L}$,M,β, and ϵ.

Within the ShortVectors algorithm, we will assume that the samples generated over βC (line 3) are exactly uniform. By doing this, we claim the probability that ShortVectors returns a $(1+\epsilon)$-approximate solution to the SAP problem by changes by at most $2^{-(n+1)}$. To see this, note that we generate exactly $\mathbf{N}_0$ such samples, all of which are η-uniform. Therefore by Lemma 1, we have that the total variation distance between the vector of approximately uniform samples and truly uniform samples is at most $\mathbf{N}_0\eta = 2^{-(n+1)}$. Lastly, the event that ShortVectors returns $(1+\epsilon)$-approximate solution is a random function of these samples, and hence when switching uniform samples for η-uniform ones, the probability of this event changes by at most $2^{-(n+1)}$. Therefore to prove the theorem, it suffices to show that the failure probability under truly uniform samples is at most $2^{-(n+1)}$.

In the proof, we adopt all the names of parameters and variables defined in the execution of ShortVectors. We denote the pairs at stage t as $(X_1^t, Y_1^t), \ldots, (X_{N_t}^t, Y_{N_t}^t)$. We also let $C_{\mathbf{v}}^+, C_{\mathbf{v}}^-$ be as in Lemma 5. For any stage $t \geq 0$, we define the pair (X_i^t, Y_i^t), $i \in [\mathbf{N}_t]$, as *good* if $X_i^t \in \text{int}(C_{\mathbf{v}}^+) \cup \text{int}(C_{\mathbf{v}}^-)$.

We now outline the remaining analysis, which we break up into four separate claims. We note that at a high level, the analysis is identical to the ones presented in [13,14].

- **Claim 1:** With high probability, the number of *good* pairs sampled at stage 0 is *large*.

  The argument is a straightforward Chernoff bound.

- **Claim 2:** After the main sieving stage (while loop lines 5-7), we still have many *good* pairs remaining.

  Using the packing argument from Lemma 5 we argue that we only remove $2(5/\gamma)^n$ vectors at each stage, and hence even if all the removed vectors are *good*, there will still be many *good* pairs left at the last stage.

- **Claim 3:** At the last stage T, for any *good* pair (X_i^T, Y_i^T), we can randomly (using a carefully chosen distribution) send it to $(X_i^T \pm \mathbf{v}, Y_i^T)$ without changing the global output distribution of the stage T variables.

Here we use a symmetry $F_{\mathbf{v}}$ (see equation (8) for details) of the base sampling distribution over $\beta(C \cup -C)$ which preserves cosets of $\mathcal{L}$, i.e. $F_{\mathbf{v}}(sZ) \equiv_D sZ$ and $F_{\mathbf{v}}(\mathbf{x}) \pmod B = \mathbf{x} \pmod B$. Here the main idea is that since the algorithm doesn't inspect the X_i^T coordinate of a surviving pair (X_i^T, Y_i^T) during the main sieving stage (other than knowing its coset representative), one can apply the symmetry $F_{\mathbf{v}}$ via $(X_i^T, Y_i^T) \to (F_{\mathbf{v}}(X_i^T), Y_i^T)$ without changing the global output distribution.

– **Claim 4:** At the last stage T, the difference set

$$\{Y_i^T - X_i^T - (Y_j^T - X_j^T) : \ i, j \in [\mathbf{N}_t]\} \setminus M$$

contains a lattice vector $\mathbf{w}$ close to $\mathbf{v}$ w.h.p.

Let $(X_i^T, Y_i^T), i \in [N_G]$ be the *good* pairs at the last stage T. The sieve allows us to conclude (with a slight technical modification) that all the lattice vectors $\{Y_i^T - X_i^T : i \in [N_G]\}$ are of length $\leq 4\beta$ (i.e. short). Using a packing bound, we group these lattice vectors into at most $O(1/\gamma\epsilon)^n$ clusters (substantially fewer than the number of *good* vectors) of radius $\beta\epsilon$. We use the symmetry $F_{\mathbf{v}}$ to send each *good* pair $Y_i^T - X_i^T \to Y_i - F_{\mathbf{v}}(X_i) = Y_i^T - X_i^T \pm \mathbf{v}$ (this is legal in the analysis since it doesn't change the global output distribution). We now argue that w.h.p. at least one pair of lattice vectors $Y_i^T - F_{\mathbf{v}}(X_i)^T$, $Y_j^T - F_{\mathbf{v}}(X_j)^T$ in the same cluster have difference equal to $\pm\mathbf{v} + \text{error}$, where the error has length $\leq \beta\epsilon$. This follows from the closeness of the lattice vectors in the same cluster and the random $\pm v$ "jiggling" induced by the $F_{\mathbf{v}}$ map.

We now formalize the above claims and give the full proofs.

Claim 1: Let $\mathcal{G}$ denote the event that there are at least $\frac{1}{2}\left(\frac{\gamma}{4}\right)^n \mathbf{N}_0$ *good* pairs at stage 0. Then $\mathcal{G}$ occurs with probability least $1 - e^{-\frac{1}{48}\gamma^n \mathbf{N}_0}$.

Proof Let $G_i = I[X_i^0 \in \text{int}(C_{\mathbf{v}}^+) \cup \text{int}(C_{\mathbf{v}}^-)]$ for $i \in [\mathbf{N}_0]$ denote the indicator random variables denoting whether (X_i^0, Y_i^0) is *good* or not. Let s_i, $i \in [\mathbf{N}_0]$, denote the $\{-1, 1\}$ random variable indicating whether X_i^0 is sampled uniformly from βC or $-\beta C$. Since $\beta \leq \|v\|_C \leq \frac{3}{2}\beta$, by Lemma 7 we have that

$$\begin{aligned}\Pr[G_i = 1] &\geq \sum_{b \in \{-1,+1\}} \Pr(X_i^0 \in \text{int}(C_{\mathbf{v}}^b) | s_i = b) \Pr(s_i = b) \\ &= \frac{1}{2}\frac{\text{vol}_n(\beta C \cap (\mathbf{v} - \beta C))}{\text{vol}_n(\beta C)} + \frac{1}{2}\frac{\text{vol}_n((\beta C - \mathbf{v}) \cap (-\beta C))}{\text{vol}_n(-\beta C)} \geq \left(\frac{\gamma}{4}\right)^n\end{aligned}$$

From the above we see that $E[\sum_{i=1}^N G_i] \geq \left(\frac{\gamma}{4}\right)^n \mathbf{N}_0$. Since the G_i's are iid Bernoulli random variables, by the Chernoff bound we get that $\Pr[\mathcal{G}] = \Pr[\sum_{i=1}^N G_i < \frac{1}{2}\left(\frac{\gamma}{4}\right)^n \mathbf{N}_0] \leq e^{-\frac{1}{48}\gamma^n \mathbf{N}_0}$, as needed.

Claim 2: Let T denote the last stage of the sieve (i.e. value of t at end of the while loop). Then conditioned on $\mathcal{G}$, the number of *good* pairs at stage T is at least $N_G = 4\left(\frac{9}{\gamma\epsilon}\right)^n$.

Proof Examine (X_i^0, Y_i^0) for $i \in [\mathbf{N}_0]$. We first claim that $\|Y_i^0\|_C^* \leq D$. To see this note that $Y_i^0 = B\mathbf{z}$ where $\mathbf{z} = B^{-1}X_i^0 - \lfloor B^{-1}X_i^0 \rfloor \in [0,1)^n$. Hence

$$\|Y_i^0\|_C^* \leq \|Y_i^0\|_C = \|\sum_{j=1}^n \mathbf{b}_j \mathbf{z}_j\|_C \leq \sum_{j=1}^n \mathbf{z}_j \|\mathbf{b}_j\|_C \leq n \max_{1\leq j\leq n} \|\mathbf{b}_j\|_C = D$$

as needed. Let $D_t = \max\{\|Y_i^t\|_C^* : i \in [\mathbf{N}_t]\}$, where we note that above shows that $D_0 \leq D$. By Lemma 5, we know that $\mathbf{N}_t \geq \mathbf{N}_{t-1} - 2\left(\frac{5}{\gamma}\right)^n$ and that $D_t \leq \frac{1}{2}D_{t-1} + \beta$ for $t \geq 1$. For $D_t \geq 3\beta$, we see that $\frac{1}{2}D_t + \beta \leq \frac{5}{6}D_t$. Given the previous bounds, an easy computation reveals that $T \leq \lceil \frac{\ln(\frac{D}{3\beta})}{\ln(\frac{6}{5})} \rceil \leq \lceil 6\ln(\frac{D}{\beta}) \rceil$.

From the above, we see that during the entire sieving phase we remove at most $T(2)\left(\frac{5}{\gamma}\right)^n \leq 2\lceil 6\ln(\frac{D}{\beta})\rceil \left(\frac{5}{\gamma}\right)^n$ pairs. Since we never modify the X_i^t's during the sieving operation, any pair that starts off as *good* stays *good* as long as it survives through the last stage. Since we start with at least $\frac{1}{2}\left(\frac{\gamma}{4}\right)^n \mathbf{N}_0$ *good* pairs in stage 0, we are left with at least

$$\frac{1}{2}\left(\frac{\gamma}{4}\right)^n \mathbf{N}_0 - 2\lceil 6\ln\left(\frac{D}{\beta}\right)\rceil \left(\frac{5}{\gamma}\right)^n \geq 4\left(\frac{9}{\gamma\epsilon}\right)^n = N_G$$

good pairs at stage T as required.

Modifying the output: Here we will analyze a way of modifying the output of ShortVectors, which maintains the global output distribution but makes the output analysis far simpler.

Let $w(\mathbf{x}) = I[\mathbf{x} \in \beta C] + I[\mathbf{x} \in -\beta C]$. Letting Z be uniform in βC and s be uniform in $\{-1, 1\}$ (i.e. the distribution in line 3), for $\mathbf{x} \in \beta(C \cup -C)$ we have that the probability density function of sZ is

$$\begin{aligned} \mathrm{d}\Pr[sZ = \mathbf{x}] &= \mathrm{d}\Pr[Z = \mathbf{x}]\Pr[s = 1] + \mathrm{d}\Pr[Z = -\mathbf{x}]\Pr[s = -1] \\ &= \frac{1}{2}\left(\frac{I[\mathbf{x} \in \beta C]}{\mathrm{vol}_n(\beta C)} + \frac{I[\mathbf{x} \in -\beta C]}{\mathrm{vol}_n(\beta C)}\right) = \frac{w(\mathbf{x})}{2\mathrm{vol}_n(\beta(C))} \ . \end{aligned} \tag{7}$$

Examine the function $f_{\mathbf{v}} : \beta(C \cup -C) \to \beta(C \cup -C)$ defined by

$$f_{\mathbf{v}}(\mathbf{x}) = \begin{cases} \mathbf{x} - \mathbf{v} : & \mathbf{x} \in \mathrm{int}(C_{\mathbf{v}}^+) \\ \mathbf{x} + \mathbf{v} : & \mathbf{x} \in \mathrm{int}(C_{\mathbf{v}}^-) \\ \mathbf{x} : & \text{otherwise} \end{cases}$$

Since $\mathrm{int}(C_{\mathbf{v}}^+) \cap \mathrm{int}(C_{\mathbf{v}}^-) = \emptyset$, it is easy to see that $f_{\mathbf{v}}$ is a well-defined bijection on $\beta(C \cup -C)$ satisfying $f_{\mathbf{v}}(f_{\mathbf{v}}(\mathbf{x})) = \mathbf{x}$. Furthermore by construction, we see that $f_{\mathbf{v}}(\mathrm{int}(C_{\mathbf{v}}^+)) = \mathrm{int}(C_{\mathbf{v}}^-)$ and $f_{\mathbf{v}}(\mathrm{int}(C_{\mathbf{v}}^-)) = \mathrm{int}(C_{\mathbf{v}}^+)$. Lastly, note that for

any $\mathbf{x} \in \beta(C \cup -C)$, that $f_{\mathbf{v}}(\mathbf{x}) \equiv \mathbf{x} \pmod{B}$ since $f_{\mathbf{v}}(\mathbf{x})$ is just a lattice vector shift of $\mathbf{x}$.

We note that if a sample X_i^0, $i \in [N_0]$, is *good* then $f_{\mathbf{v}}(X_i^0) = X_i^0 \pm \mathbf{v}$, and if it is *not good* then $f_{\mathbf{v}}(X_i^0) = X_i^0$. This is the main property we will need about good "perturbations".

Let $F_{\mathbf{v}}$ denote the random function where

$$F_{\mathbf{v}}(\mathbf{x}) = \begin{cases} \mathbf{x} & : \text{with probability } \frac{w(\mathbf{x})}{w(\mathbf{x})+w(f_{\mathbf{v}}(\mathbf{x}))} \\ f_{\mathbf{v}}(\mathbf{x}) & : \text{with probability } \frac{w(f_{\mathbf{v}}(\mathbf{x}))}{w(\mathbf{x})+w(f_{\mathbf{v}}(\mathbf{x}))} \end{cases} \tag{8}$$

Here, we intend that different applications of the function $F_{\mathbf{v}}$ all occur with independent randomness. Crucially, we will see that $F_{\mathbf{v}}$ defines a symmetry of the base sample distribution over $\beta(C \cup -C)$, i.e. $F_{\mathbf{v}}(X_i^0) \equiv_D sZ$ (see equation (10) below). Next we define the function $c_{\mathbf{v}}$ as

$$c_{\mathbf{v}}(\mathbf{x}, \mathbf{y}) = \begin{cases} f_{\mathbf{v}}(\mathbf{x}) & : \|\mathbf{y} - f_{\mathbf{v}}(\mathbf{x})\|_C^* < \|\mathbf{y} - \mathbf{x}\|_C^* \\ \mathbf{x} & : \text{otherwise} \end{cases}$$

The sole purpose of the $c_{\mathbf{v}}$ function is to be able to send a *good* last stage pair (X_i^T, Y_i^T) to a representative for which $Y_i^T - X_i^T$ is short under $\|\cdot\|_C^*$. This will be crucial for enabling the packing bound in Claim 4.

We claim that for any $\mathbf{x}, \mathbf{y}, \mathbf{v}$ that $F_{\mathbf{v}}(\mathbf{x}) \equiv_D F_{\mathbf{v}}(c_{\mathbf{v}}(\mathbf{x}, \mathbf{y}))$. This follows from equation (8), noting that

$$\begin{aligned} \Pr[F_{\mathbf{v}}(\mathbf{x}) = \mathbf{x}] = \Pr[F_{\mathbf{v}}(c_{\mathbf{v}}(\mathbf{x}, \mathbf{y})) = \mathbf{x}] &= \frac{w(\mathbf{x})}{w(\mathbf{x}) + w(f_{\mathbf{v}}(\mathbf{x}))} \\ \Pr[F_{\mathbf{v}}(\mathbf{x}) = f_{\mathbf{v}}(\mathbf{x})] = \Pr[F_{\mathbf{v}}(c_{\mathbf{v}}(\mathbf{x}, \mathbf{y})) = f_{\mathbf{v}}(\mathbf{x})] &= \frac{w(f_{\mathbf{v}}(\mathbf{x}))}{w(\mathbf{x}) + w(f_{\mathbf{v}}(\mathbf{x}))} \end{aligned} \tag{9}$$

We now prove that $F_{\mathbf{v}}$ yields a symmetry of distribution of sZ:

$$\begin{aligned} \mathrm{d}\Pr[F_{\mathbf{v}}(sZ) = \mathbf{x}] &= \mathrm{d}\Pr[sZ = \mathbf{x}] \Pr[F_{\mathbf{v}}(\mathbf{x}) = \mathbf{x}] + \mathrm{d}\Pr[sZ = f_v(\mathbf{x})] \Pr[F_{\mathbf{v}}(f_{\mathbf{x}}(\mathbf{x})) = \mathbf{x}] \\ &= \frac{w(\mathbf{x})}{2\mathrm{vol}_n(\beta C)} \frac{w(\mathbf{x})}{w(\mathbf{x}) + w(f_{\mathbf{v}}(\mathbf{x}))} + \frac{w(f_{\mathbf{v}}(\mathbf{x}))}{2\mathrm{vol}_n(\beta C)} \frac{w(\mathbf{x})}{w(\mathbf{x}) + w(f_{\mathbf{v}}(\mathbf{x}))} \\ &= \frac{w(\mathbf{x})}{2\mathrm{vol}_n(\beta C)} = \mathrm{d}\Pr[sZ = \mathbf{x}] \quad (\text{ by equation (7) }) \end{aligned} \tag{10}$$

For any stage $t \geq 0$, and $i \in [\mathbf{N}_t]$, define $\bar{X}_i^t = c_{\mathbf{v}}(X_i^t, Y_i^t)$.

Claim 3: For any stage $t \geq 0$, the pairs $(X_1^t, Y_1^t), \ldots, (X_{\mathbf{N}_t}^t, Y_{\mathbf{N}_t}^t)$, and $(F_{\mathbf{v}}(\bar{X}_1^t), Y_1^t), \ldots, (F_{\mathbf{v}}(\bar{X}_{\mathbf{N}_t}^t), Y_{\mathbf{N}_t}^t)$ are identically distributed. Furthermore, this remains true after conditioning on the event $\mathcal{G}$.

Proof To prove the claim, it suffices to show that the sequences

$$1.\ (X_1^t, Y_1^t), \ldots, (X_{\mathbf{N}_t}^t, Y_{\mathbf{N}_t}^t) \quad \text{and} \quad 2.\ (F_{\mathbf{v}}(\bar{X}_1^t), Y_1^t), (X_2^t, Y_2^t), \ldots, (X_{\mathbf{N}_t}^t, Y_{\mathbf{N}_t}^t)$$

are identically distributed both before and after conditioning on $\mathcal{G}$. Our analysis will be independent of the index analyzed, hence the claim will follow by applying the proof inductively on each remaining pair in the second list.

First, we begin by showing that the sequences

$$2.\ (F_{\mathbf{v}}(\bar{X}_1^t), Y_1^t), \ldots, (X_{N_t}^t, Y_{N_t}^t) \quad \text{and} \quad 3.\ (F_{\mathbf{v}}(X_1^t), Y_1^t), \ldots, (X_{\mathbf{N}_t}^t, Y_{\mathbf{N}_t}^t)$$

are identically distributed. Let us first condition on any valid instantiation of the normal stage t variables,

$$(X_1^t, Y_1^t), \ldots, (X_{\mathbf{N}_t}, Y_{\mathbf{N}_t}) = (\mathbf{x}_1^t, \mathbf{y}_1^t), \ldots, (\mathbf{x}_{N_t}^t, \mathbf{y}_{N_t}^t).$$

From equation (9), we know that as distributions $F_{\mathbf{v}}(c_{\mathbf{v}}(\mathbf{x}_1^t, \mathbf{y}_1^t)) \equiv_D F_{\mathbf{v}}(\mathbf{x}_1^t)$, where we remember that $\bar{\mathbf{x}}_1^t = c_{\mathbf{v}}(\mathbf{x}_1^t, \mathbf{y}_1^t)$. Therefore sequences 2 and 3 are clearly identically distributed. Since the above equivalence holds for any full conditioning of the stage t variables, we see that the equivalence must also hold in general, and in particular before or after conditioning on $\mathcal{G}$.

It now remains to show that the sequences

$$1.\ (X_1^t, Y_1^t), \ldots, (X_{\mathbf{N}_t}^t, Y_{\mathbf{N}_t}^t) \quad \text{and} \quad 3.\ (F_{\mathbf{v}}(X_1^t), Y_1^t), \ldots, (X_{\mathbf{N}_t}^t, Y_{\mathbf{N}_t}^t)$$

are identically distributed. The pairs in (1) correspond exactly to the induced distribution of the algorithm on the stage t variables. We think of the pairs in (3) as the induced distribution of a modified algorithm on these variables, where the modified algorithm just runs the normal algorithm and replaces (X_1^t, Y_1^t) by $(F_{\mathbf{v}}(X_1^t), Y_1^t)$ in stage t. To show the distributional equivalence, we show a probability preserving correspondence between runs of the normal and modified algorithm having the same stage t variables.

For $0 \leq k \leq t$, let the pairs $(\mathbf{x}_i^k, \mathbf{y}_i^k)$, $i \in [N_k]$, denote a valid run of the normal algorithm through stage t. We label this as run A. Let us denote the sequence of ancestors of $(\mathbf{x}_1^t, \mathbf{y}_1^t)$ in the normal algorithm by $(\mathbf{x}_{a_k}^k, \mathbf{y}_{a_k}^k)$ for $0 \leq k < t$. By definition of this sequence, we have that $\mathbf{x}_{a_0}^0 = \mathbf{x}_{a_1}^1 = \cdots = \mathbf{x}_1^t$. Since the ShortVectors algorithm is deterministic given the initial samples, the probability density of this run is simply

$$\mathrm{d}\Pr\left[\cap_{i \in [\mathbf{N}_0]} \{X_i^0 = \mathbf{x}_i^0\}\right] = \mathrm{d}\Pr\left[X_{a_0}^0 = \mathbf{x}_1^t\right] \prod_{i \in [\mathbf{N}_0], i \neq a_0} \mathrm{d}\Pr\left[X_i^0 = \mathbf{x}_i^0\right] \quad (11)$$

by the independence of the samples and since $\mathbf{x}_{a_0}^0 = \mathbf{x}_1^t$. Notice if we condition on the event $\mathcal{G}$, assuming the run A belongs to $\mathcal{G}$ (i.e. that there are enough *good* pairs at stage 0), the above probability density is simply divided by $\Pr[\mathcal{G}]$.

If $\mathbf{x}_1^t \notin \mathrm{int}(C_{\mathbf{v}}^+) \cup \mathrm{int}(C_{\mathbf{v}}^-)$, note that $F_{\mathbf{v}}(\mathbf{x}_1^t) = \mathbf{x}_1^t$, i.e. the action of $F_{\mathbf{v}}$ is trivial. In this case, we associate run A with the identical run for the modified algorithm, which is clearly valid and has the same probability. Now assume that $\mathbf{x}_1^t \in C_{\mathbf{v}}^+ \cup C_{\mathbf{v}}^-$. In this case, we associate run A to two runs of the modified

algorithm: $\tilde{A}$, identical to run A, and C, run A with $(\mathbf{x}_{a_k}^k, \mathbf{y}_{a_k}^k)$ replaced by $(f_{\mathbf{v}}(\mathbf{x}_{a_k}^k), \mathbf{y}_{a_k}^k)$ for $0 \leq k < t$. Note that both of the associated runs have the same stage t variables as run A by construction.

We must check that both runs are indeed valid for the modified algorithm. To see this, note that up till stage t, the modified algorithm just runs the normal algorithm. Run $\tilde{A}$ inherents validity for these stages from the fact that run A is valid for the normal algorithm. To see that run C is valid, we first note that $f_{\mathbf{v}}(\mathbf{x}_{a_0}^0) \equiv \mathbf{x}_{a_0}^0 \equiv \mathbf{y}_{a_0}^0 \pmod{B}$ (B is the lattice basis), which gives validity for stage 0. By design of the normal sieving algorithm, note that during run A, the algorithm never inspects the contents of $\mathbf{x}_{a_k}^k$ for $0 \leq k < t$. This is because none of the $\mathbf{y}_{a_k}^k$, $0 \leq k < t$, is designated as a cluster center during the runs of the Sieving Procedure. Therefore, if $(\mathbf{x}_{a_k}^k, \mathbf{y}_{a_k}^k)$ denotes a valid ancestor sequence in run A, then so does $(f_{\mathbf{v}}(\mathbf{x}_{a_k}), \mathbf{y}_{a_k}^k)$ in run C for $0 \leq k < t$. For stage t, note that the normal algorithm, given the stage 0 inputs of run $\tilde{A}$ would output $(\mathbf{x}_1^t, \mathbf{y}_1^t), \ldots, (\mathbf{x}_{N_t}^t, \mathbf{y}_{N_t}^t)$ for the stage t variables, and that given the stage 0 inputs of run C would output $(f_{\mathbf{v}}(\mathbf{x}_1^t), \mathbf{y}_1^t), (\mathbf{x}_2^t, \mathbf{y}_2^t), \ldots, (\mathbf{x}_{N_t}^t, \mathbf{y}_{N_t}^t)$. Hence in run $\tilde{A}$, the modified algorithm retains the normal algorithms output, and in run C, it swiches it from $f_{\mathbf{v}}(\mathbf{x}_1^t)$ back to $\mathbf{x}_1^t$. Therefore, both runs are indeed valid for the modified algorithm. Furthermore, note that if run A is in $\mathcal{G}$, then both the stage 0 variables of $\tilde{A}$ and C index a *good* run for the normal algorithm since the pair $(\mathbf{x}_{a_0}^0, \mathbf{y}_{a_0})$ is *good* iff $(f_{\mathbf{v}}(\mathbf{x}_{a_0}^0), \mathbf{y}_{a_0})$ is *good*. Hence we see that correspondence described is valid both before and after conditioning on $\mathcal{G}$. Lastly, it is clear that for any run of the modified algorithm, the above correspondence yields a unique run of the normal algorithm.

It remains to show that the correspondence is probability preserving. We must therefore compute the probability density associated with the union of run $\tilde{A}$ and C for the modified algorithm. Using the analysis from the previous paragraph and the computation in (11), we see that this probability density is

$$\begin{aligned} \Big(\mathrm{d}\Pr[X_{a_0}^0 = \mathbf{x}_1^t]\Pr[F_{\mathbf{v}}(\mathbf{x}_1^t) = \mathbf{x}_1^t] \quad + \\ \mathrm{d}\Pr[X_{a_0}^0 = f_{\mathbf{v}}(\mathbf{x}_1^t)]\Pr[F_{\mathbf{v}}(f_{\mathbf{v}}(\mathbf{x}_1^t)) = \mathbf{x}_1^t]\Big) \prod_{i \in [\mathbf{N}_0], i \neq a_0} d\Pr\left[X_i^0 = \mathbf{x}_i^0\right] \end{aligned} \tag{12}$$

On the first line above, the first term corresponds to run $\tilde{A}$ which samples $\mathbf{x}_1^t$ in stage 0 and then chooses to keep $(\mathbf{x}_1^t, \mathbf{y}_1^t)$ in stage t, and the second term corresponds to C which samples $f_{\mathbf{v}}(\mathbf{x}_1^t)$ in stage 0 and chooses to flip $(f_{\mathbf{v}}(\mathbf{x}_1^t), \mathbf{y}_1^t)$ to $(\mathbf{x}_1^t, \mathbf{y}_1^t)$ in stage t. Now by definition of $F_{\mathbf{v}}$ and since $F_{\mathbf{v}}(X_{a_0}^0) \equiv_D X_{a_0}^0$ (see equation (10)), we have that

$$\begin{aligned} \Big(\mathrm{d}\Pr[X_{a_0}^0 = \mathbf{x}_1^t]\Pr[F_{\mathbf{v}}(\mathbf{x}_1^t) = \mathbf{x}_1^t] + \mathrm{d}\Pr[X_{a_0}^0 = f_{\mathbf{v}}(\mathbf{x}_1^t)]\Pr[F_{\mathbf{v}}(f_{\mathbf{v}}(\mathbf{x}_1^t)\Big) \\ = \Pr[F_{\mathbf{v}}(X_{a_0}^0) = \mathbf{x}_1^t] = \Pr[X_{a_0}^0 = \mathbf{x}_1^t]. \end{aligned}$$

Hence the probabilities in equations (11) and (12) are equal as needed. Lastly, note that when conditioning on $\mathcal{G}$, both of the corresponding probabilities are divided by $\Pr[\mathcal{G}]$, and hence equality is maintained.

Output Analysis:

Claim 4: Let T denote the last stage of the sieve. Then conditioned on the event $\mathcal{G}$, with probability at least $1 - \left(\frac{2}{3}\right)^{\frac{1}{2}N_G}$ there exists a lattice vector $\mathbf{w}$ in the set $\{Y_i^T - X_i^T - (Y_j^T - X_j^T) : i, j \in [\mathbf{N}_T]\}$ satisfying

$$(\dagger) \quad \mathbf{w} \in \mathcal{L} \setminus M, \ \mathbf{w} - \mathbf{v} \in M \cap \mathcal{L} \text{ and } \|\mathbf{w} - \mathbf{v}\|_C < \epsilon\beta.$$

Furthermore, any lattice vector satisfying (†) is a $(1+\epsilon)$-approximate solution to SAP.

Proof Let $(\mathbf{x}_1^T, \mathbf{y}_1^T), \ldots, (\mathbf{x}_{N_T}^T, \mathbf{y}_{N_T}^T)$ denote any valid instantiation of the stage T variables corresponding to a *good* run of the algorithm (i.e. one belonging to $\mathcal{G}$). Let $(\bar{\mathbf{x}}_i^T, \mathbf{y}_i^T) = (c_{\mathbf{v}}(\mathbf{x}_i^T, \mathbf{y}_i^T), \mathbf{y}_i^T)$ for $i \in [N_T]$. By claim 3, it suffices to prove the claim for the pairs $(F_{\mathbf{v}}(\bar{\mathbf{x}}_1^T), \mathbf{y}_1^T), \ldots, (F_{\mathbf{v}}(\bar{\mathbf{x}}_{N_T}^T), \mathbf{y}_{N_T}^T)$. This follows since the probability of success (i.e. the existence of the desired vector) conditioned on $\mathcal{G}$, is simply the average over all instantiations above of the conditional probability of success.

Since our instantiation corresponds to a *good* run, by Claim 2 we have at least N_G *good* pairs in stage T. Since $c_{\mathbf{v}}$ preserves *good* pairs, the same holds true for $(\bar{\mathbf{x}}_i^T, \mathbf{y}_i^T)$, $i \in [N_T]$. For notational convenience, let us assume that the pairs $(\bar{\mathbf{x}}_i^T, \mathbf{y}_i^T)$, $i \in [N_G]$ are all *good*. We remember that $f_{\mathbf{v}}(\bar{\mathbf{x}}_i^T) = \bar{\mathbf{x}}_i^T \pm \mathbf{v}$ and $f_{\mathbf{v}}(f_{\mathbf{v}}(\bar{\mathbf{x}}_i^T)) = \bar{\mathbf{x}}_i^T$ for $i \in [N_G]$.

First, since T is last the stage, we know that $\|\mathbf{y}_i^T\|_C^* \leq 3\beta$ for $i \in [N_G]$. Next, for $i \in [N_G]$, by definition of $c_{\mathbf{v}}$ we have that

$$\|\mathbf{y}_i^T - \bar{\mathbf{x}}_i^T\|_C^* = \min\{\|\mathbf{y}_i^T - \mathbf{x}_i^T\|_C^*, \|\mathbf{y}_i^T - f_{\mathbf{v}}(\mathbf{x}_i^T)\|_C^*\}$$

Let $s = s(\mathbf{y}_i^T)$, i.e. $\|\mathbf{y}_i^T\|_C^* = \|\mathbf{y}_i^T\|_{sC}$. Since $(\mathbf{x}_i^T, \mathbf{y}_i^T)$ is *good* at least one of $-\mathbf{x}_i^T, -f_{\mathbf{v}}(\mathbf{x}_i^T) \in \beta sC$. Without loss of generality, we assume $-\mathbf{x}_i^T \in \beta sC$. Therefore, we get that

$$\begin{aligned} \|\mathbf{y}_i^T - \bar{\mathbf{x}}_i^T\|_C^* &\leq \|\mathbf{y}_i^T - \mathbf{x}_i^T\|_C^* \leq \|\mathbf{y}_i^T - \mathbf{x}_i^T\|_{sC} \\ &\leq \|\mathbf{y}_i^T\|_{sC} + \| - \mathbf{x}_i^T\|_{sC} \leq 3\beta + \beta = 4\beta \end{aligned} \tag{13}$$

Let S denote the set $\{\mathbf{y}_i^T - \bar{\mathbf{x}}_i^T : i \in [N_G]\}$. Since $\bar{\mathbf{x}}_i^T \equiv \mathbf{y}_i^T \pmod{B}$, we note that $S \subseteq \mathcal{L}$. Also by equation (13) we have that $S \subseteq 4\beta(C \cup -C) \cap \mathcal{L}$. Let $\Lambda \subseteq S$ denote a maximal subset such that

$$\left(\mathbf{x} + \frac{\epsilon}{2}\beta\text{int}(C \cap -C)\right) \cap \left(\mathbf{y} + \frac{\epsilon}{2}\beta\text{int}(C \cap -C)\right) = \emptyset$$

for distinct $\mathbf{x}, \mathbf{y} \in \Lambda$. Since $S \subseteq 4\beta(C \cup -C)$, we see that for $\mathbf{x} \in S$

$$\mathbf{x} + \frac{\epsilon}{2}\beta(C \cap -C) \subseteq 4\beta(C \cup -C) + \frac{\epsilon}{2}\beta(C \cap -C) \subseteq \left(4 + \frac{\epsilon}{2}\right)\beta(C \cup -C)$$

Therefore we see that

$$|\Lambda| \leq \frac{\mathrm{vol}_n((4+\frac{\epsilon}{2})\beta(C \cup -C))}{\mathrm{vol}_n(\frac{\epsilon}{2}\ \beta(C \cap -C))} = \left(\frac{8+\epsilon}{\epsilon}\right)^n \frac{\mathrm{vol}_n(C \cup -C)}{\mathrm{vol}_n(C \cap -C)}$$
$$\leq \left(\frac{8+\epsilon}{\epsilon}\right)^n \frac{2\mathrm{vol}_n(C)}{\gamma^n \mathrm{vol}_n(C)} \leq 2\left(\frac{9}{\gamma\epsilon}\right)^n \leq \frac{1}{2}N_G$$

Since Λ is maximal, we note that for any $\mathbf{x} \in S$, there exists $\mathbf{y} \in \Lambda$ such that

$$\mathrm{int}(\mathbf{x} + \frac{\epsilon}{2}\beta(C \cap -C)) \cap \mathrm{int}(\mathbf{y} + \frac{\epsilon}{2}\beta(C \cap -C)) \neq \emptyset \Leftrightarrow \|\mathbf{x} - \mathbf{y}\|_{C \cap -C} < \epsilon\beta \quad (14)$$

Let $c_1, \ldots, c_{|\Lambda|} \in [N_G]$, denote indices such that $\Lambda = \{\mathbf{y}_{c_i}^T - \bar{\mathbf{x}}_{c_i}^T : 1 \leq i \leq |\Lambda|\}$, and let $C = \{c_j : 1 \leq j \leq |\Lambda|\}$. For $j \in \{1, \ldots, |\Lambda|\}$, recursively define the sets

$$I_j = \{i \in [N_G] : \|(\mathbf{y}_i^T - \bar{\mathbf{x}}_i^T) - (\mathbf{y}_{c_j}^T - \bar{\mathbf{x}}_{c_j}^T)\|_{C \cap -C} < \epsilon\beta\} \setminus \left(C \ \cup \ (\cup_{k=1}^{j-1} I_k)\right) \quad (15)$$

Given equation (14), we have by construction that the sets $C, I_1, \ldots, I_{|\Lambda|}$ partition $[N_G]$. For each $j \in \{1, \ldots, |\Lambda|\}$, we examine the differences

$$S_j = \pm\{(\mathbf{y}_i^T - F_{\mathbf{v}}(\bar{\mathbf{x}}_i^T)) - (\mathbf{y}_{c_j}^T - F_{\mathbf{v}}(\bar{\mathbf{x}}_{c_j}^T)) : i \in I_j\}$$

We will show that S_j fails to contain a vector satisfying (†) with probability at most $\left(\frac{2}{3}\right)^{|I_j|}$. First we note that $S_j \subseteq \mathcal{L}$ since $\mathbf{y}_i^T \equiv \bar{\mathbf{x}}_i^T \equiv F_{\mathbf{v}}(\bar{\mathbf{x}}_i^T) \pmod{B}$ for $i \in [N_G]$.

We first condition on the value of $F_{\mathbf{v}}(\bar{\mathbf{x}}_{c_j}^T)$ which is either $\bar{\mathbf{x}}_{c_j}^T, \bar{\mathbf{x}}_{c_j}^T - \mathbf{v}$ or $\bar{\mathbf{x}}_{c_j}^T + \mathbf{v}$. We examine the case where $F_{\mathbf{v}}(\bar{\mathbf{x}}_{c_j}^T) = \bar{\mathbf{x}}_{c_j}^T$, the analysis for the other two cases is similar. Now, for $i \in I_j$, we analyze the difference

$$\mathbf{y}_i^T - F_{\mathbf{v}}(\bar{\mathbf{x}}_i^T) - (\mathbf{y}_{c_j}^T - \bar{\mathbf{x}}_{c_j}^T) \quad (16)$$

Let $\delta_i = (\mathbf{y}_i^T - \bar{\mathbf{x}}_i^T) - (\mathbf{y}_{c_j}^T - \bar{\mathbf{x}}_{c_j}^T)$. Depending on the output of $F_{\mathbf{v}}(\bar{\mathbf{x}}_i^T)$, note that the vector (16) is either (a) δ_i or of the form (b) $\pm\mathbf{v} + \delta_i$ (since $f_{\mathbf{v}}(\bar{\mathbf{x}}_i^T) = \bar{\mathbf{x}}_i^T \pm \mathbf{v}$). We claim that a vector of form (b) satisfies (†). To see this, note that after possibly negating the vector, it can be brought to the form $\mathbf{v} \pm \delta_i \in \mathcal{L}$, where we have that

$$\|\pm\delta_i\|_C < \|\pm\delta_i\|_{C \cap -C} = \|\delta_i\|_{C \cap -C} < \epsilon\beta \leq \epsilon\lambda(C, \mathcal{L}, M) < \lambda(C, \mathcal{L}, M), \quad (17)$$

since $i \in I_j$ and $\epsilon \leq \frac{1}{2}$. Since $\delta_i \in \mathcal{L}$ and $\|\delta_i\|_C < \lambda(C, \mathcal{L}, M)$, we must have that $\pm\delta_i \in M \cap \mathcal{L}$. Next, since $\mathbf{v} \in \mathcal{L} \setminus M$ and $\pm\delta_i \in M$, we have that $\mathbf{v} \pm \delta_i \in \mathcal{L} \setminus M$. Lastly, note that

$$\|\mathbf{v} \pm \delta_i\|_C \leq \|\mathbf{v}\|_C + \|\pm\delta_i\|_C < \lambda(C, \mathcal{L}, M) + \epsilon\beta \leq (1+\epsilon)\lambda(C, \mathcal{L}, M)$$

as required.

Now the probability the vector in (16) is of form (b) is

$$\Pr[F_{\mathbf{v}}(\bar{\mathbf{x}}_i^T) = f_{\mathbf{v}}(\bar{\mathbf{x}}_i^T)] = \frac{w(f_{\mathbf{v}}(\bar{\mathbf{x}}_i^T))}{w(\bar{\mathbf{x}}_i^T) + w(f_{\mathbf{v}}(\bar{\mathbf{x}}_i^T))} \geq \frac{1}{3}$$

since for any $\mathbf{x} \in \beta(C \cup -C)$ we have that $1 \leq w(\mathbf{x}) \leq 2$. Since each $i \in I_j$ indexes a vector in S_j not satisfying (†) with probability at most $1 - \frac{1}{3} = \frac{2}{3}$, the probability that S_j contains no vector satisfying (†) is at most $\left(\frac{2}{3}\right)^{|I_r|}$ (by independence) as needed.

Let F_j, $j \in \{1, \ldots, |\Lambda|\}$, denote the event that S_j does not contain a vector satisfying (†). Note that F_j only depends on the pairs $(F_{\mathbf{v}}(\bar{\mathbf{x}}_{c_j}^T), \mathbf{y}_{c_j}^T)$ and $(F_{\mathbf{v}}(\bar{\mathbf{x}}_i^T), \mathbf{y}_i^T)$ for $i \in I_j$. Since the sets $I_1, \ldots, I_{|\Lambda|}, C$ partition $[N_G]$, these dependencies are all disjoint, and hence the events are independent. Therefore the probability that none of $S_1, \ldots, S_{|\Lambda|}$ contains a vector satisfying (†) is at most

$$\Pr[\cap_{j=1}^{|\Lambda|} F_j] \leq \prod_{j=1}^{|\Lambda|} \left(\frac{2}{3}\right)^{|I_j|} = \left(\frac{2}{3}\right)^{N_G - |\Lambda|} \leq \left(\frac{2}{3}\right)^{\frac{1}{2}N_G}$$

as needed.

Runtime and Failure Probability: We first analyze the runtime. First, we make $O(n \log \frac{R}{r})$ guesses for the value of $\Lambda(C, \mathcal{L}, M)$. We run the ShortVectors algorithm once for each such guess β. During one iteration of the sieving algorithm, we first generate $\mathbf{N}_0$ η-uniform samples from βC (line 3). By Theorem 6, this takes $\text{poly}(n, \ln \frac{1}{\eta}, \ln \beta, \ln R, \ln r)$ time per sample. We also mod each sample by the basis B for $\mathcal{L}$, which takes $\text{poly}(|B|)$ time ($|B|$ is the bit size of the basis). Next, by the analysis of Claim 2, we apply the sieving procedure at most $\lceil 6 \ln \frac{D}{\beta} \rceil$ times (runs of while loop at line 5), where each iteration of the sieving procedure (line 6) takes at most $O(\mathbf{N}_0 \left(\frac{5}{\gamma}\right)^n)$ time by Lemma 5. Lastly, we return the set of differences (line 8), which takes at most $O(\mathbf{N}_0^2)$ time. Now by standard arguments, one has that the values D and β (for each guess) each have size (bit description length) polynomial in the input, i.e. polynomial in $|B|$ (bit size of the basis of $\mathcal{L}$), n, $\ln R$, $\ln r$. Since $\mathbf{N}_0 = O(\ln(\frac{D}{\beta})(\frac{36}{\gamma^2\epsilon})^n)$, we have that the total running time is

$$\text{poly}(n, \ln R, \ln r, |B|, \frac{1}{\epsilon})\ O\left(\frac{1}{\gamma^4\epsilon^2}\right)^n \text{ as needed.}$$

Furthermore, the space usage is the algorithm clearly controlled by the space needed to store the sample pairs $(X_1^0, Y_1^0), \ldots, (X_{\mathbf{N}_0}^0, Y_{\mathbf{N}_0}^0)$. Hence the space usage is bounded by $\text{poly}(n, |B|, \ln R, \ln r)\mathbf{N}_0 = \text{poly}(n, |B|, \ln R, \ln r)2^{O(n)}(\frac{1}{\gamma^2\epsilon})^n$ as needed.

We now analyze the success probability. Here we only examine the guess β, where $\beta \leq \lambda(C, \mathcal{L}, M) \leq \frac{3}{2}\beta$. Assuming perfectly uniform samples over βC, by the analysis of Claim 4, we have that conditioned on $\mathcal{G}$, we fail to output a $(1+\epsilon)$ approximate solution to SAP with probability at most $\left(\frac{2}{3}\right)^{\frac{1}{2}N_G}$. Hence, under the uniform sampling assumption, the total probability of failure is at most

$$\left(\frac{2}{3}\right)^{\frac{1}{2}N_G} + \Pr[\mathcal{G}^c] \leq \left(\frac{2}{3}\right)^{\frac{1}{2}N_G} + e^{-\frac{1}{48}\gamma^n\mathbf{N}_0} \ll 2^{-(n+1)}$$

by Claim 2. When switching to η-uniform samples, as argued in the preliminary analysis, this failure probability increases by at most the total variation distance, i.e. by at most $\eta \mathbf{N}_0 = 2^{-(n+1)}$. Therefore, the algorithm succeeds with probability at least $1 - 2^{-(n+1)} - 2^{-(n+1)} = 1 - 2^{-n}$ as needed.

We remark that in the above analysis, if we assume that the samples are exactly uniform, then the error probability becomes *doubly exponentially* small. Indeed, if one is willing to let the uniform sampler over C run for exponential time (allowing for η to be doubly exponentially small), then the total error probability can be reduced to this level. This in fact improves over the analyses of [13,14], where they even with exact samples the error probability remains exponentially small. Here, we achieve this improvement by a better analysis of the pairwise difference set in the last step.

Exact SAP: Here we are given the guarantee that $\lambda(C, \mathcal{L}, M) \leq t\lambda_1(C, \mathcal{L})$, and we wish to use our SAP solver to get an exact minimizer to the SAP. To solve this, we run the approximate SAP solver on $C, \mathcal{L}, M$ with parameter $\epsilon = \frac{1}{t}$, which takes $2^{O(n)}(t^2/\gamma^4)^n$-time and $2^{O(n)}(t/\gamma^2)^n$-space. Let $\mathbf{v} \in \mathcal{L} \setminus M$ be a lattice vector satisfying $\|\mathbf{v}\|_C = \lambda(C, \mathcal{L}, M)$. By Claim 4, with probability at least $1 - 2^{-n}$ we are guaranteed to output a lattice vector $\mathbf{w} \in \mathcal{L} \setminus M$, such that

$$\|\mathbf{w} - \mathbf{v}\|_C < \epsilon\lambda(C, \mathcal{L}, M) \leq \left(\frac{1}{t}\right) t\lambda_1(C, \mathcal{L}) = \lambda_1(C, \mathcal{L})$$

However, since $\mathbf{w} - \mathbf{v} \in \mathcal{L}$ and $\|\mathbf{w} - \mathbf{v}\|_C < \lambda_1(C, \mathcal{L})$, we must have that $\mathbf{w} - \mathbf{v} = 0$. Therefore $\mathbf{w} = \mathbf{v}$ and our SAP solver returns an exact minimizer as needed.

3.3 Closest Vector Problem

In this section, we present a reduction from Approximate-CVP to Approximate-SAP for general norms. In [13], it is shown that ℓ_p CVP reduces to ℓ_p SAP in one higher dimension. By relaxing the condition that the lifted SAP problem remain in ℓ_p, we give a very simple reduction which reduces CVP in any norm to SAP in one higher dimension under a different norm that is essentially as symmetric. Given the generality of our SAP solver, such a reduction suffices.

Theorem 9 (Approximate-CVP) *Take $\mathbf{x} \in \mathbb{R}^n$. Then for any $\epsilon \in (0, \frac{1}{3})$, $\mathbf{y} \in \mathcal{L}$ satisfying $\|\mathbf{y} - \mathbf{x}\|_C \leq (1+\epsilon)d_C(\mathcal{L}, \mathbf{x})$ can be computed in time $O(\frac{1}{\gamma^4\epsilon^2})^n$ with probability at least $1 - 2^{-n}$. Furthermore, if $d_C(\mathcal{L}, \mathbf{x}) \leq t\lambda_1(C, \mathcal{L})$, $t \geq 2$, then a vector $\mathbf{y} \in \mathcal{L}$ satisfying $\|\mathbf{y} - \mathbf{x}\|_C = d_C(\mathcal{L}, \mathbf{x})$ can be computed in time $O(\frac{t^2}{\gamma^4})^n$ with probability at least $1 - 2^{-n}$.*

Proof To show the theorem, we use a slightly modified version of Kannan's lifting technique to reduce CVP to SAP. Let us define $\mathcal{L}' \subseteq \mathbb{R}^{n+1}$ as the lattice generated by $(\mathcal{L}, 0)$ and $(-\mathbf{x}, 1)$.

In the standard way, we first guess a value $\beta > 0$ satisfying $\beta \leq d_C(\mathcal{L}, \mathbf{x}) \leq \frac{3}{2}\ \beta$. Now let $C' = C \times [-\frac{1}{2\beta}, \frac{1}{\beta}]$. For $(\mathbf{y}, z)$, $\mathbf{y} \in \mathbb{R}^n$, $z \in \mathbb{R}$, we have that

$$\|(\mathbf{y}, z)\|_{C'} = \max\{\|\mathbf{y}\|_C, \beta z, -2\beta z\}$$

Also, note that $C' \cap -C' = (C \cap -C) \times [-\frac{1}{2\beta}, \frac{1}{2\beta}]$. Now we see that $\mathrm{vol}_{n+1}(C' \cap -C') = \frac{1}{\beta}\mathrm{vol}_n(C \cap -C)$ and $\mathrm{vol}_{n+1}(C') = \frac{3}{(2\beta)}\mathrm{vol}_n(C)$. Therefore

$$\mathrm{vol}_{n+1}(C' \cap -C') = \frac{1}{\beta}\mathrm{vol}_n(C \cap -C) \geq \frac{1}{\beta}\ \gamma^n \mathrm{vol}_n(C) = \frac{2}{3}\ \gamma^n \mathrm{vol}_{n+1}(C').$$

Hence C' is $\gamma^{\frac{n}{n+1}}(\frac{2}{3})^{\frac{1}{n+1}} \geq \gamma(1-\frac{1}{n})$ symmetric. Let $M = \{\mathbf{y} \in \mathbb{R}^{n+1} : \mathbf{y}_{n+1} = 0\}$. Define $m : \mathcal{L} \to \mathcal{L}' \setminus M$ by $m(\mathbf{y}) = (\mathbf{y} - \mathbf{x}, 1)$, where it is easy to see that m is well-defined and injective. Define

$$S = \{\mathbf{y} \in \mathcal{L} : \|\mathbf{y} - \mathbf{x}\|_C \leq (1+\epsilon) d_C(\mathcal{L}, \mathbf{x})\} \quad \text{and}$$
$$S' = \{\mathbf{y} \in \mathcal{L}' \setminus M : \|\mathbf{y}\|'_C \leq (1+\epsilon)\lambda(C', \mathcal{L}', M)\}.$$

We claim that m defines a norm preserving bijection between S and S'. Taking $\mathbf{y} \in \mathcal{L}$, we see that

$$\|m(\mathbf{y})\|_{C'} = \|(\mathbf{y} - \mathbf{x}, 1)\|_{C'} = \max\{\|\mathbf{y} - \mathbf{x}\|_C, \beta, -2\beta\} = \|\mathbf{y} - \mathbf{x}\|_C$$

since $\beta \leq d_C(\mathcal{L}, \mathbf{x}) \leq \|\mathbf{y} - \mathbf{x}\|_C$ by construction. So we have that $\|m(\mathbf{y})\|_{C'} = \|\mathbf{y} - \mathbf{x}\|_C$, and hence $\lambda(C', \mathcal{L}', M) \leq \inf_{\mathbf{y} \in \mathcal{L}} \|\mathbf{y} - \mathbf{x}\|_C = d_C(\mathcal{L}, \mathbf{x})$. Next take $(\mathbf{y}, z) \in \mathcal{L}' \setminus M$, $\mathbf{y} \in \mathbb{R}^n$, $z \in \mathbb{R}$, such that $\|(\mathbf{y}, z)\|_{C'} \leq (1+\epsilon)\lambda(C', \mathcal{L}', M)$. We claim that $z = 1$. Assume not, then since $(\mathbf{y}, z) \in \mathcal{L}' \setminus M$, we must have that either $z \geq 2$ or $z \leq -1$. In either case, we have that

$$\|(\mathbf{y}, z)\|_{C'} = \max\{\|\mathbf{y}\|_C, \beta z, -2\beta z\} \geq \max\{\beta z, -2\beta z\} \geq 2\beta$$

Now since $\beta \leq d_C(\mathcal{L}, \mathbf{x}) \leq \frac{3}{2}\ \beta$, $\epsilon \in (0, \frac{1}{3})$, and that $\lambda(C', \mathcal{L}', M) \leq d_C(\mathcal{L}, \mathbf{x})$, we get that

$$\begin{aligned}\|(\mathbf{y}, z)\|_{C'} \geq 2\beta = (1 + \frac{1}{3})(\frac{3}{2}\beta) &\geq (1 + \frac{1}{3}) d_C(\mathcal{L}, \mathbf{x}) \\ &> (1+\epsilon) d_C(\mathcal{L}, \mathbf{x}) \geq (1+\epsilon)\lambda(C', \mathcal{L}', M)\end{aligned}$$

a clear contradiction to our initial assumption. Since $z = 1$, we may write $\mathbf{y} = \mathbf{w} - \mathbf{x}$ where $\mathbf{w} \in \mathcal{L}$. Therefore, we see that

$$\|(\mathbf{y}, z)\|_{C'} = \|(\mathbf{w} - \mathbf{x}, 1)\|_{C'} = \max\{\|\mathbf{w} - \mathbf{x}\|_C, \beta, -2\beta\} = \|\mathbf{w} - \mathbf{x}\|_C$$

since $\|\mathbf{w} - \mathbf{x}\|_C \geq d_C(\mathcal{L}, \mathbf{x}) \geq \beta$. So we have that $(1+\epsilon)\lambda(C', \mathcal{L}', M) \geq \|(\mathbf{y}, z)\|_{C'} = \|\mathbf{w} - \mathbf{x}\|_C \geq d_C(\mathcal{L}, \mathbf{x})$. Since the previous statement still holds when choosing $\epsilon = 0$, we must have that $\lambda(C', \mathcal{L}', M) \geq d_C(\mathcal{L}, \mathbf{x})$ and hence $\lambda(C', \mathcal{L}', M) = d_C(\mathcal{L}, \mathbf{x})$.

From the above, for $\mathbf{y} \in S$, we have that $\|m(\mathbf{y})\|_{C'} = \|\mathbf{y} - \mathbf{x}\|_C \leq (1 + \epsilon) d_C(\mathcal{L}, \mathbf{x}) = (1+\epsilon)\lambda(C', \mathcal{L}', M)$, and hence $m(\mathbf{y}) \in S'$ as needed. Next if $(\mathbf{y}, z) \in S'$, from the above we have that $z = 1$, and hence $(\mathbf{y}, z) = (\mathbf{w} - \mathbf{x}, 1)$

where $\mathbf{w} \in \mathcal{L}$. Therefore $(\mathbf{y}, z) = m(\mathbf{w})$, where $\|\mathbf{w} - \mathbf{x}\|_C = \|(\mathbf{y}, z)\|_{C'} \leq (1+\epsilon)\lambda(C', \mathcal{L}', M) = (1+\epsilon)d_C(\mathcal{L}, x)$, and hence $w \in S$. Since the map m is injective, we get that m defines a norm preserving bijection between S and S' as claimed.

Hence solving $(1+\epsilon)$-CVP with respect to $C, \mathcal{L}, \mathbf{x}$ is equivalent to solving $(1+\epsilon)$-SAP with respect to $C', \mathcal{L}', M$. Hence applying the $1+\epsilon$ approximation algorithm for SAP from Theorem 8, we get an algorithm for $(1+\epsilon)$-CVP which runs in $2^{O(n)}(\frac{1}{\gamma^4\epsilon^2})^n$-time and $2^{O(n)}(\frac{1}{\gamma^2\epsilon})^n$-space and succeeds with probability at least $1 - 2^{-n}$ as required.

For exact CVP, we are given the guarantee that $d_C(\mathcal{L}, \mathbf{x}) \leq t\lambda_1(C, \mathcal{L})$. From analysis above, we see that

$$\lambda_1(C', \mathcal{L}') = \min\{\lambda_1(C', \mathcal{L}', M), \inf_{\mathbf{y} \in \mathcal{L} \setminus \{0\}} \|(\mathbf{y}, 0)\|_{C'}\} = \min\{d_C(\mathcal{L}, \mathbf{x}), \lambda_1(C, \mathcal{L})\}$$

Therefore

$$\begin{aligned} \lambda(C', \mathcal{L}', M) &= d_C(\mathcal{L}, \mathbf{x}) = \min\{d_C(\mathcal{L}, \mathbf{x}), t\lambda_1(C, \mathcal{L})\} \\ &\leq t \min\{d_C(\mathcal{L}, \mathbf{x}), \lambda_1(C, \mathcal{L})\} = t\lambda_1(C', \mathcal{L}') \end{aligned}$$

Hence we may again use the SAP solver in Theorem 8 to solve the exact CVP problem in $2^{O(n)}(\frac{t^2}{\gamma^4})^n$-time and $2^{O(n)}(\frac{t}{\gamma^2})^n$-space with probability at least $1 - 2^{-n}$ as required.

4 Acknowledgments

I would like to thank Santosh Vempala for useful discussions relating to this problem.

References

1. Ralph Gomory. An outline of an algorithm for solving integer programs. *Bulletin of the American Mathematical Society*, 64(5):275–278, 1958.
2. Hendrik W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.
3. Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of operations research*, 12(3):415–440, 1987.
4. Robert Hildebrand and Matthias Köppe. A new lenstra-type algorithm for quasiconvex polynomial integer minimization with complexity $2^{O(n \log n)}$. Arxiv, Report 1006.4661, 2010. `http://arxiv.org`.
5. Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. *SIAM Journal of Computing*, 42(3):1364–1391, 2013. Preliminary version in STOC 2010.
6. Daniel Dadush, Chris Peikert, and Santosh Vempala. Enumerative lattice algorithms in any norm via M-ellipsoid coverings. In *Proceedings of 52nd annual Symposium on Foundations of Computer Science (FOCS)*, 580–589, 2011.
7. Alexander Barvinok. A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed. *Mathematics of Operations Research*, 19(4):769–779, 1994.

8. Ravi Kannan. Test sets for integer programs, $\forall\exists$ sentences. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science Volume 1*, 39–47, 1990.
9. Friedrich Eisenbrand and Gennady Shmonin. Parametric integer programming in fixed dimension. *Mathematics of Operations Research*, 33(4):839–850, 2008.
10. Sebastian Heinz. Complexity of integer quasiconvex polynomial optimization. *Journal of Complexity*, 21(4):543–556, 2005. Festschrift for the 70th Birthday of Arnold Schonhage.
11. Miklós Ajtai, Ravi Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings of 33rd Symposium on the Theory of Computing (STOC)*, 601–610, 2001.
12. Miklós Ajtai, Ravi Kumar, and D. Sivakumar. Sampling short lattice vectors and the closest lattice vector problem. In *Proceedings of the 17th Conference on Computational Complexity (CCC)*, 53–57, 2002.
13. Johannes Blömer and Stefanie Naewe. Sampling methods for shortest vectors, closest vectors and successive minima. *Theoretical Computer Science*, 410(18):1648–1665, 2009. Preliminary version in ICALP 2007.
14. Vikraman Arvind and Pushkar S. Joglekar. Some sieving algorithms for lattice problems. In *Proceedings of 28th Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, 25–36, 2008.
15. Friedrich Eisenbrand, Nicolai Hähnle, and Martin Niemeier. Covering cubes and the closest vector problem. In *Proceedings of the 27th annual ACM symposium on Computational Geometry (SoCG)*, 417–423, 2011.
16. Aleksandr Y. Khinchin. A quantitative formulation of Kronecker's theory of approximation. *Izv. Acad. Nauk SSSR*, 12:113–122, 1948.
17. László Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986. Preliminary version in STACS 1985.
18. Jeffrey C. Lagarias, Hendrik W. Lenstra Jr., and Claus-Peter Schnorr. Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice. *Combinatorica*, 10(4):333–348, 1990.
19. Ravi Kannan and László Lovász. Covering minima and lattice point free convex bodies. *Annals of Mathematics*, 128:577–602, 1988.
20. Wojciech Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296:625–635, 1993.
21. Wojciech Banaszczyk. Inequalities for convex bodies and polar reciprocal lattices in R^n II: Application of k-convexity. *Discrete and Computational Geometry*, 16:305–311, 1996.
22. Wojciech Banaszczyk, Alexander Litvak, Alain Pajor, and Stanislaw Szarek. The flatness theorem for nonsymmetric convex bodies via the local theory of Banach spaces. *Mathematics of Operations Research*, 24(3):728–750, 1999.
23. Mark Rudelson. Distance between non-symmetric convex bodies and the MM*-estimate. *Positivity*, 4(8):161–178, 2000.
24. Oded Goldreich, Daniele Micciancio, Shmuel Safra, and Jean-Pierre Seifert. Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. *Information Processing Letters*, 71(2):55–61, 1999.
25. Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, 1988.
26. Martin E. Dyer, Alan M. Frieze, and Ravi Kannan. A random polynomial time algorithm for approximating the volume of convex bodies. In *Proceedings of the 21st Symposium on the Theory of Computing (STOC)*, 375–381, 1989.
27. Vitali D. Milman and Alain Pajor. Entropy and asymptotic geometry of non-symmetric convex bodies. *Advances in Mathematics*, 152(2):314–335, 2000.
28. Ravi Kannan, László Lovász, and Miklós Simonovits. Isoperimetric problems for convex bodies and a localization lemma. *Discrete & Computational Geometry*, 13:541–559, 1995.
29. Grigoris Paouris. Concentration of mass on isotropic convex bodies. *Comptes Rendus Mathematique*, 342(3):179–182, 2006.

A Appendix

A.1 Preliminaries

Logconcave functions: A function $f : \mathbb{R}^n \to \mathbb{R}_+$ is logconcave if for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, $0 \leq \alpha \leq 1$, we have that $f(\mathbf{x})^\alpha f(\mathbf{y})^{1-\alpha} \leq f(\alpha\mathbf{x} + (1-\alpha)\mathbf{y})$. For a convex body $K \subseteq \mathbb{R}^n$, the indicator function I_K of K is easily seen to be logconcave. A logconcave density $f : \mathbb{R}^n \to \mathbb{R}_+$ is a logconcave function for which $\int_{\mathbb{R}^n} f(\mathbf{x})d\mathbf{x} = 1$.

A logconcave random variable $X \in \mathbb{R}^n$ is logconcave if X admits a logconcave density $f : \mathbb{R}^n \to \mathbb{R}_+$. We note that the uniform distribution over K admits a logconcave density, i.e. $\pi_K(\mathbf{x}) = \frac{1}{\mathrm{vol}_n(K)} I_K[\mathbf{x}]$ for $\mathbf{x} \in \mathbb{R}^n$. A classical fact is that for two logconcave random variables $X, Y \in \mathbb{R}^n$, the sum $X + Y$ is also a logconcave random variable. The random variable X (with density f) is isotropic if $\mathrm{E}[X] = \int_{\mathbb{R}^n} \mathbf{x} f(\mathbf{x})d\mathbf{x} = \mathbf{0}$ (mean zero), and if $\mathrm{E}[XX^t] = \left(\int_{\mathbb{R}^n} \mathbf{x}_i\mathbf{x}_j f(\mathbf{x})d\mathbf{x}\right)_{ij} = I_n$ (covariance matrix identity), the $n \times n$ identity. For any full-dimensional random variable $X \in \mathbb{R}^n$, there exists an affine transformation $T : \mathbb{R}^n \to \mathbb{R}^n$ (unique up to rotations) such that TX is isotropic. A convex body K is isotropic if the uniform distribution over K is isotropic.

We will need the following two theorems in the proof of Lemma 2. The first theorem due to Kannan, Lovász and Simonovits gives sandwiching estimates for isotropic convex bodies.

Theorem 10 (Isotropic Sandwiching [28]) *Let $K \subseteq \mathbb{R}^n$ be an isotropic convex body. Then*

$$\sqrt{\frac{n+2}{n}} \cdot B_2^n \subseteq K \subseteq \sqrt{n(n+2)} \cdot B_2^n.$$

The following theorem of Paouris gives strong concentration estimates for logconcave random variables.

Theorem 11 (Measure Concentration [29], Theorem 11) *Let $X \in \mathbb{R}^n$ be an isotropic logconcave random variable. Then for some absolute constant $c > 0$,*

$$\Pr[\|X\| \geq c\sqrt{n}t] \leq e^{-\sqrt{n}t}$$

for all $t \geq 1$.

Proof (Lemma 2: Approx. Barycenter) Let $X_1, \ldots, X_N$ denote iid uniform samples over $K \subseteq \mathbb{R}^n$, where $N = \left(\frac{2c}{\epsilon}\right)^2 n$. We will show that for $\mathbf{b} = \frac{1}{N}\sum_{i=1}^n X_i$, that the following holds

$$\Pr[\| \pm (\mathbf{b} - \mathbf{b}(K))\|_{K-\mathbf{b}(K)} > \epsilon] \leq 4^{-n} \tag{18}$$

Since the above statement is invariant under affine transformations, we may assume K is isotropic, i.e. $\mathbf{b}(K) = \mathrm{E}[X_1] = \mathbf{0}$, the origin, and $\mathrm{E}[X_1X_1^t] = I_n$, the $n \times n$ identity. Since K is isotropic, from Theorem 10 we have that $B_2^n \subseteq K$. Therefore to show (18) it suffices to prove that $\Pr[\|\mathbf{b}\|_2 > \epsilon] \leq 4^{-n}$. Since the X_i's are iid isotropic random vectors, we see that $\mathrm{E}[\mathbf{b}] = \frac{1}{N}\sum_{i=1}^N \mathrm{E}[X_i] = 0$ and

$$\mathrm{E}[\mathbf{b}\mathbf{b}^t] = \frac{1}{N^2} \sum_{i,j \in [N]} E[X_iX_j^t] = \frac{1}{N^2} \sum_{i=1}^n E[X_iX_i^t] = \frac{1}{N} I_n$$

Now since the X_is are logconcave, we have that $\mathbf{b}$ is also logconcave. Note that the random variable $\sqrt{N}\mathbf{b}$ is logconcave and isotropic, since $\mathrm{E}[\sqrt{N}\mathbf{b}(\sqrt{N}\mathbf{b})^t] = N\,\mathrm{E}[\mathbf{b}\mathbf{b}^t] = I_n$. Therefore, by the concentration inequality of Paouris 11 we have that

$$\Pr[\|\mathbf{b}\|_2 > \epsilon] = \Pr[\|\sqrt{N}\mathbf{b}\|_2 > \sqrt{N}\epsilon] = \Pr[\|\sqrt{N}\mathbf{b}\|_2 > 2c\sqrt{n}] < e^{-2n} < 4^{-n}$$

as claimed. To prove the theorem, we note that when switching the X_i's from truly uniform to 4^{-n} uniform, the above probability changes by at most $n\left(\frac{2c}{\epsilon^2}\right)4^{-n}$ by Lemma 1. Therefore the total error probability under 4^{-n}-uniform samples is at most 2^{-n} as needed.

Convexity:

Proof (Lemma 3: Estimates for norm recentering) We have $\mathbf{z} \in \mathbb{R}^n$, $\mathbf{x}, \mathbf{y} \in K$ satisfying (†) $\| \pm (\mathbf{x} - \mathbf{y})\|_{K-\mathbf{y}} \leq \alpha < 1$. We prove the statements as follows:

1. $\|\mathbf{z} - \mathbf{y}\|_{K-\mathbf{y}} \leq \tau \Leftrightarrow (\mathbf{z} - \mathbf{y}) \in \tau(K - \mathbf{y}) \Leftrightarrow \mathbf{z} \in \tau K + (1 - \tau)\mathbf{y}$ as needed.
2. Let $\tau = \|\mathbf{z} - \mathbf{x}\|_{K-\mathbf{x}}$. Then by (1), we have that $\mathbf{z} \in \tau K + (1 - \tau)\mathbf{x}$. Now note that

$$(1 - \tau)(\mathbf{x} - \mathbf{y}) \subseteq |1 - \tau|\alpha(K - \mathbf{y})$$

by assumption (†) and (1). Therefore

$$\begin{aligned}\mathbf{z} \in \tau K + (1 - \tau)\mathbf{x} &= \tau K + (1 - \tau)\mathbf{y} + (1 - \tau)(\mathbf{x} - \mathbf{y}) \subseteq \tau K + (1 - \tau)\mathbf{y} + \alpha|1 - \tau|(K - \mathbf{y}) \\ &= (\tau + \alpha|1 - \tau|)K + (1 - \tau - \alpha|1 - \tau|)\mathbf{y}\end{aligned}$$

Hence by (1), we have that

$$\|\mathbf{z} - \mathbf{y}\|_{K-y} \leq \tau + \alpha|1 - \tau| = \|\mathbf{z} - \mathbf{x}\|_{K-\mathbf{x}} + \alpha|1 - \|\mathbf{z} - \mathbf{x}\|_{K-\mathbf{x}}|$$

as needed.

3. We first show that

$$\pm(\mathbf{y} - \mathbf{x}) \in \frac{\alpha}{1 - \alpha}(K - \mathbf{x})$$

By (1) and (†) we have that

$$\begin{aligned}(\mathbf{x} - \mathbf{y}) \in \alpha(K - \mathbf{y}) &\Leftrightarrow (\mathbf{x} - \mathbf{y}) - \alpha(\mathbf{x} - \mathbf{y}) \in \alpha(K - \mathbf{y}) - \alpha(\mathbf{x} - \mathbf{y}) \\ &\Leftrightarrow (1 - \alpha)(\mathbf{x} - \mathbf{y}) \in \alpha(K - \mathbf{x}) \Leftrightarrow (\mathbf{x} - \mathbf{y}) \in \frac{\alpha}{1 - \alpha}(K - \mathbf{x})\end{aligned}$$

as needed. Next since $0 \leq \alpha \leq 1$, we have that $|1 - 2\alpha| \leq 1$. Therefore by (†) we have that

$$(1 - 2\alpha)(\mathbf{y} - \mathbf{x}) \in |1 - 2\alpha|\alpha(K - \mathbf{y}) \subseteq \alpha(K - \mathbf{y})$$

since $0 \in K - y$. Now note that

$$\begin{aligned}(1 - 2\alpha)(\mathbf{y} - \mathbf{x}) \in \alpha(K - \mathbf{y}) &\Leftrightarrow (1 - 2\alpha)(\mathbf{y} - \mathbf{x}) + \alpha(\mathbf{y} - \mathbf{x}) \in \alpha(K - \mathbf{y}) + \alpha(\mathbf{y} - \mathbf{x}) \\ &\Leftrightarrow (1 - \alpha)(\mathbf{y} - \mathbf{x}) \in \alpha(K - \mathbf{x}) \Leftrightarrow (\mathbf{y} - \mathbf{x}) \in \frac{\alpha}{1 - \alpha}(K - \mathbf{x})\end{aligned}$$

as needed.
Let $\tau = \|\mathbf{z} - \mathbf{y}\|_{K-\mathbf{y}}$. Then by (1), we have that $\mathbf{z} \in \tau K + (1 - \tau)\mathbf{y}$. Now note that

$$\begin{aligned}\mathbf{z} \in \tau K + (1 - \tau)\mathbf{y} &= \tau K + (1 - \tau)\mathbf{x} + (1 - \tau)(\mathbf{y} - \mathbf{x}) \\ &\subseteq \tau K + (1 - \tau)\mathbf{x} + \frac{\alpha}{1 - \alpha}|1 - \tau|(K - \mathbf{x}) \\ &= (\tau + \frac{\alpha}{1 - \alpha}|1 - \tau|)K + (1 - \tau - \frac{\alpha}{1 - \alpha}|1 - \tau|)\mathbf{x}\end{aligned}$$

Hence by (1), we have that

$$\|\mathbf{z} - \mathbf{x}\|_{K-\mathbf{x}} \leq \tau + \frac{\alpha}{1 - \alpha}|1 - \tau| = \|\mathbf{z} - \mathbf{y}\|_{K-\mathbf{y}} + \frac{\alpha}{1 - \alpha}|1 - \|\mathbf{z} - \mathbf{y}\|_{K-\mathbf{y}}|$$

as needed.

Proof (Corollary 1: Stability of symmetry)
We claim that $(1 - \|\mathbf{x}\|_K)(K \cap -K) \subseteq (K - \mathbf{x}) \cap (\mathbf{x} - K)$. Take $\mathbf{z} \in K \cap -K$, then note that

$$\begin{aligned}\|\mathbf{x} + (1 - \|\mathbf{x}\|_K)\mathbf{z}\|_K &\leq \|\mathbf{x}\|_K + (1 - \|\mathbf{x}\|_K)\|\mathbf{z}\|_K \leq \|\mathbf{x}\|_K + (1 - \|\mathbf{x}\|_K)\|\mathbf{z}\|_{K \cap -K} \\ &\leq \|\mathbf{x}\|_K + (1 - \|\mathbf{x}\|_K) = 1\end{aligned}$$

hence $\mathbf{x} + (1 - \|\mathbf{x}\|_K)(K \cap -K) \subseteq K \Leftrightarrow (1 - \|\mathbf{x}\|_K)(K \cap -K) \subseteq K - \mathbf{x}$. Next note that

$$\begin{aligned}\| - \mathbf{x} + (1 - \|\mathbf{x}\|_K)\mathbf{z}\|_{-K} &\leq \| - \mathbf{x}\|_{-K} + (1 - \|\mathbf{x}\|_K)\|\mathbf{z}\|_{-K} \leq \|\mathbf{x}\|_K + (1 - \|\mathbf{x}\|_K)\|\mathbf{z}\|_{K\cap -K} \\ &\leq \|\mathbf{x}\|_K + (1 - \|\mathbf{x}\|_K) = 1\end{aligned}$$

hence $-\mathbf{x} + (1 - \|\mathbf{x}\|_K)(K \cap -K) \subseteq -K \Leftrightarrow (1 - \|\mathbf{x}\|_K)(K \cap -K) \subseteq \mathbf{x} - K$, as needed. Now we see that

$$\mathrm{vol}_n((K - \mathbf{x}) \cap (\mathbf{x} - K)) \geq \mathrm{vol}_n((1 - \|\mathbf{x}\|_K)(K \cap -K)) = (1 - \|\mathbf{x}\|_K)^n \mathrm{vol}_n(K \cap -K)$$

and so the claim follows from Theorem 7.

A.2 Integer Programming

Proof (Lemma 4: Well-Centered Bodies) Clearly for any $\mathbf{a}'_0 \in K_{a,b}$ we have that $K_{a,b} \subseteq \mathbf{a}'_0 + 2RB_2^n$, since $K_{a,b} \subseteq K \subseteq \mathbf{a}_0 + RB_2^n$. Therefore the we need only worry that $K_{a,b}$ contains a polynomially sized ball around an easy to compute point in $K_{a,b}$. Since a, b correspond to m, u in the while loop (lines 6-13 in Algorithm 1), we have that

$$b - a \geq \frac{\delta}{2} \qquad \langle \mathbf{x}_l, \mathbf{v}\rangle + \frac{3\delta}{16} \leq b \qquad a \leq \langle \mathbf{x}_u, \mathbf{v}\rangle - \frac{7\delta}{16},$$

where the last inequality follows since $a + \frac{\delta}{2} \leq b \leq \langle \mathbf{x}_u, \mathbf{v}\rangle + \frac{\delta}{16}$. Since $\mathbf{a}_0 + rB_2^n \subseteq K$, by assumption $\delta \leq \frac{1}{64} r\|\mathbf{v}\|_2$, we also have that

$$\mathrm{width}_K(\mathbf{v}) \geq \mathrm{width}_{\mathbf{a}_0 + rB_2^n}(\mathbf{v}) = 2r\|\mathbf{v}\|_2 \geq 128\delta$$

From the above, we additionally conclude that

$$\langle \mathbf{x}_l, \mathbf{v}\rangle \leq \langle \mathbf{a}_0, \mathbf{v}\rangle \leq \langle \mathbf{x}_u, \mathbf{v}\rangle \quad \text{and} \quad \langle \mathbf{x}_u, \mathbf{v}\rangle - \langle \mathbf{x}_l, \mathbf{v}\rangle \geq \mathrm{width}_K(\mathbf{v}) - \frac{\delta}{8} \geq 127\delta$$

Let I denote the interval $[\langle \mathbf{x}_l, \mathbf{v}\rangle, \langle \mathbf{x}_u, \mathbf{v}\rangle] \cap [a, b]$. Combining the above inequalities, it is not hard to check that $\mathrm{length}(I) \geq \frac{3\delta}{16}$ (corresponding to having b shifted as far to the left as possible). Let $I_l = [\langle \mathbf{x}_l, \mathbf{v}\rangle, \langle \mathbf{x}, \mathbf{a}_0\rangle] \cap I$ and $I_u = [\langle \mathbf{a}_0, \mathbf{v}\rangle, \langle \mathbf{x}_u, \mathbf{v}\rangle] \cap I$. Since I_l and I_u partition I, we must have that either $\mathrm{length}(I_l) \geq \frac{3\delta}{32}$ or $\mathrm{length}(I_u) \geq \frac{3\delta}{32}$. Assume we are in the former case (the analysis for the latter case is symmetric). Let c denote the midpoint of I_l. Define $\mathbf{a}'_0$ as

$$\mathbf{a}'_0 = \frac{\langle \mathbf{a}_0, \mathbf{v}\rangle - c}{\langle \mathbf{a}_0 - \mathbf{x}_l, \mathbf{v}\rangle}\mathbf{x}_l + \frac{c - \langle \mathbf{x}_l, \mathbf{v}\rangle}{\langle \mathbf{a}_0 - \mathbf{x}_l, \mathbf{v}\rangle}\mathbf{a}_0,$$

where we note that $\mathbf{a}'_0$ can easily be computed in polynomial time. Since $\mathbf{a}'_0$ is a convex combination of $\mathbf{x}_l$ and $\mathbf{a}_0$, and since $\langle \mathbf{a}'_0, \mathbf{v}\rangle = c \in [a, b]$, we have that $\mathbf{a}'_0 \in K_{a,b}$. Next by our assumption that $\mathrm{length}(I_l) \geq \frac{3\delta}{32}$, that c is the midpoint of I_l, and that $\|\mathbf{x}_l - \mathbf{a}_0\|_2 \leq R$, we have that

$$\frac{c - \langle \mathbf{x}_l, \mathbf{v}\rangle}{\langle \mathbf{a}_0 - \mathbf{x}_l, \mathbf{v}\rangle} \geq \frac{3\delta}{64R\|\mathbf{v}\|_2}.$$

Since $\mathbf{a}_0 + rB_2^n \subseteq K$, we get that

$$\begin{aligned}K &\supseteq \frac{\langle \mathbf{a}_0, \mathbf{v}\rangle - c}{\langle \mathbf{a}_0 - \mathbf{x}_l, \mathbf{v}\rangle}\mathbf{x}_l + \frac{c - \langle \mathbf{x}_l, \mathbf{v}\rangle}{\langle \mathbf{a}_0 - \mathbf{x}_l, \mathbf{v}\rangle}(\mathbf{a}_0 + rB_2^n) \\ &= \mathbf{a}'_0 + r\frac{c - \langle \mathbf{x}_l, \mathbf{v}\rangle}{\langle \mathbf{a}_0 - \mathbf{x}_l, \mathbf{v}\rangle}B_2^n \supseteq \mathbf{a}'_0 + \frac{3r\delta}{64R\|\mathbf{v}\|_2}B_2^n\end{aligned}$$

Furthermore, note that

$$\max\{\langle \mathbf{x}, \mathbf{v}\rangle : \mathbf{x} \in \mathbf{a}'_0 + \frac{3r\delta}{64R\|\mathbf{v}\|_2}B_2^n\} = c + \frac{3r\delta}{64R} \leq c + \frac{3\delta}{64} \leq b$$

since c is the midpoint of $I_l \subseteq [a,b]$. By the symmetric argument, we also have that $\min\{\langle \mathbf{x}, \mathbf{v}\rangle : \mathbf{x} \in \mathbf{a}'_0 + \frac{3r\delta}{64R\|\mathbf{v}\|_2} B_2^n\} \geq c - \frac{3\delta}{64} \geq a$. Therefore, we have that

$$\mathbf{a}'_0 + \frac{3r\delta}{64R\|\mathbf{v}\|_2} B_2^n \subseteq K_{a,b}$$

as needed.