Exact SVP and CVP in Structured Lattices: Polynomial-Space Algorithms and Reductions

Jiseung Kim
Jeonbuk National University
jiseungkim@jbnu.ac.kr

Changmin Lee
Korea University
changminlee@korea.ac.kr

Yongha Son
Sungshin Women's University
yongha.son@sungshin.ac.kr

Abstract

Although module lattices possess richer algebraic structure than general lattices, their algorithmic benefits for exact shortest and closest vector problems (SVP and CVP) remain relatively unexplored. For general N-dimensional lattices, the fastest known polynomial-space algorithms take $N^{\Theta(N)}$ time, whereas known $2^{O(N)}$-time algorithms require exponential space. We investigate how the algebraic structure of modules over number fields can be exploited to obtain faster exact algorithms in polynomial space.

We develop a common framework based on reduced module flags and lattice-point enumeration in Euclidean balls, with two complementary approaches. The first uses recursive subfield descent, yielding deterministic $2^{O(N)}$-time algorithms for both SVP and CVP on fixed-rank modules over totally real or CM fields with bounded root discriminant and bounded-degree subfield chains. Infinite unramified class field towers provide families of unbounded dimension satisfying these conditions. The second approach reduces lattice-point enumeration in Euclidean balls to exact SVP, without requiring a subfield chain or any bound on the root discriminant. For fixed-rank modules with root discriminant $\rho = d^{o(1)}$, where d is the field degree, this reduction yields deterministic SVP and CVP algorithms in time $d^{d/(2\mathrm{e})+o(d)}$.

Furthermore, the second approach yields exact CVP algorithms for semistable lattices, including cyclotomic ideal lattices, without any restriction on the root discriminant. The resulting running times are $N^{N/(2\mathrm{e})+o(N)}$ deterministically and $N^{N/(4\mathrm{e})+o(N)}$ with randomization. While the two approaches above initially focus on fixed-rank modules, we further extend both to modules of growing rank. In particular, for modules of rank $\Theta(\log N)$ over fields with bounded root discriminant, we obtain deterministic SVP and CVP algorithms in times $(\log N)^{N/(2\mathrm{e})+o(N)}$ and $(\log N)^{N/2+o(N)}$, respectively.

1 Introduction

The shortest vector problem (SVP) asks for a shortest nonzero lattice vector, while the closest vector problem (CVP) asks for a lattice vector nearest to a given target. For general N-dimensional lattices, known $2^{O(N)}$-time algorithms require exponential space [AKS01, MV13, ADRS15, ADS15, Hha26], whereas the fastest known polynomial-space algorithms take $N^{\Theta(N)}$ time [Kan83, HS07, FGL26].

Module lattices play a central role in lattice-based cryptography, underlying prominent post-quantum schemes such as the Kyber key-encapsulation mechanism and the Dilithium signature scheme [BDK+18, DKL+18]. Their geometry is closely related to the arithmetic properties of the underlying number field. In particular, the field discriminant has recently emerged as an important parameter in quantitative analyses of module-lattice reduction. Ducas, Engelberts, and de Perthuis [DEdP25] identify the discriminant as a key parameter governing the predicted performance of module-BKZ. These developments highlight the potential of exploiting number-field structure to improve lattice algorithms.

However, translating this algebraic structure into faster exact algorithms remains challenging. Even when the module rank is fixed, each rank-one module block has dimension proportional to the ambient lattice dimension N. Applying known general-purpose polynomial-space algorithms to such blocks still takes $N^{\Theta(N)}$ time. Moreover, exploiting a reduced module decomposition requires efficiently constructing it as well as performing the subsequent exact enumeration. This raises a natural question: can the algebraic structure of module lattices be exploited to achieve single-exponential time for exact SVP and CVP using only polynomial space?

1.1 Main Results

Building on a common framework of reduced module flags and lattice-point enumeration in Euclidean balls developed in Section 2, we obtain polynomial-space algorithms for exact SVP and CVP in several settings. We summarize our main results below.

Single-exponential algorithms via subfield chains. Our first result concerns modules over totally real or CM fields equipped with compatible rational inner products, including the canonical inner product. For a number field K, we write

$$\mathrm{rd}(K) = |\Delta_K|^{1/[K:\mathbb{Q}]}$$

for its root discriminant, where Δ_K is the field discriminant. The precise input model and compatibility condition are given in Section 2.2. Our first main result can be stated informally as follows.

Theorem 1.1 (Informal)**.** *Let M be a nonzero N-dimensional module lattice of bounded rank over a totally real or CM field K, equipped with a compatible rational inner product. Suppose that K has bounded root discriminant and admits a subfield chain with bounded relative degrees ending at a field of bounded degree.*

Then exact SVP and CVP for every rational target can be solved deterministically in $2^{O(N)}$ time and polynomial space, up to factors polynomial in the input length. The bounds include recovering the subfield chain, constructing reduced module flags, and enumerating all closest vectors. All bounded parameters are fixed independently of N.

Main algorithmic contribution. The central challenge is not merely to decompose a module lattice into rank-one blocks, but to control the cost of exact enumeration while constructing the reduced module flags required by the algorithms. In particular, the intermediate flags encountered

during construction need not be reduced. We establish uniform enumeration bounds that apply to these intermediate flags and to arbitrary rational shifts and search radii. These bounds allow the same lower-field routines to perform both flag construction and exact enumeration, yielding single-exponential total running time with polynomial working space. The formal results and proofs are given in Section 3.

A key feature of this result is that its assumptions are satisfied by infinite families of number fields of unbounded degree. Indeed, the finite layers of an infinite unramified p-class field tower have constant root discriminant and admit subfield chains of relative degrees at most p. Consequently, the single-exponential-time guarantees apply to module lattices of arbitrarily large dimension.

For a concrete example, a totally real field F of degree eight found by Martin has root discriminant less than 914 and a 2-class group of rank eight. The Golod–Shafarevich criterion then implies that F has an infinite Hilbert 2-class field tower [HMR21, Section 3.3.1]. Every finite extension K/F contained in this tower remains totally real and satisfies $\mathrm{rd}(K) = \mathrm{rd}(F) < 914$. Moreover, the 2-group structure of the tower gives a subfield chain

$$K = K_0 \supsetneq K_1 \supsetneq \cdots \supsetneq K_h = F, \qquad [K_i : K_{i+1}] = 2.$$

Thus, the root discriminant, relative degrees, and terminal field degree remain bounded as $[K : \mathbb{Q}]$ grows without bound.

The same construction also yields CM examples. Let $E = \mathbb{Q}(i)$ and replace each field K_i in the tower by K_iE. Since the original tower is everywhere unramified, so is its base change over FE. Hence

$$\mathrm{rd}(K_iE) = \mathrm{rd}(FE) \le \mathrm{rd}(F)\,\mathrm{rd}(E) < 1828.$$

Moreover, the fields K_iE are CM, with $[K_iE : K_{i+1}E] = 2$, and the terminal field FE has degree sixteen. Thus, Theorem 1.1 applies to infinite families of both totally real and CM fields.

The subfield chain need not be supplied as input: it can be recovered from the field representation (Corollary 3.7). The explicit dependence on the root discriminant is given in Theorem D.1.

An alternative approach via exact-SVP oracles. Our second approach replaces recursive subfield descent with a reduction from lattice-point enumeration in Euclidean balls to exact SVP. Unlike the first approach, this reduction requires neither a subfield chain nor a bound on the root discriminant.

For fixed-rank modules, let $d = [K : \mathbb{Q}]$ and $\rho = \mathrm{rd}(K)$. When $\rho = d^{o(1)}$, the reduction yields deterministic polynomial-space algorithms for exact SVP and CVP in time

$$d^{d/(2\mathrm{e})+o(d)}$$

(Theorem 4.1). Using the randomized polynomial-space SVP algorithm of [FGL26], we also obtain randomized algorithms running in

$$d^{d/(4\mathrm{e})+o(d)}$$

time with constant success probability (Corollary 4.10).

Exact CVP on semistable lattices. The SVP-oracle approach also yields results beyond module lattices. For arbitrary semistable lattices of dimension D, we reduce exact CVP to $2^{O(D)}$ calls to an exact-SVP oracle in dimension at most $D + 1$. This gives deterministic and randomized polynomial-space CVP algorithms with running times

$$D^{D/(2\mathrm{e})+o(D)} \qquad \text{and} \qquad D^{D/(4\mathrm{e})+o(D)},$$

respectively, with success probability at least $2/3$ for the randomized algorithm (Theorem 5.2 and Corollary 5.3).

Table 1: Comparison of exact SVP and CVP algorithms and summary of our results.

Algorithm / lattice family	**SVP time**	**CVP time**	**Space**
General lattices			
Kannan–Hanrot–Stehlé [Kan83, HS07]	$N^{N/(2\mathrm{e})+o(N)}$	$N^{N/2+o(N)}$	poly.
Feng–Gao–Liu [FGL26]†	$N^{N/(4\mathrm{e})+o(N)}$	—	poly.
Micciancio–Voulgaris [MV13]	$2^{O(N)}$	$2^{O(N)}$	exp.
Aggarwal et al. [ADRS15, ADS15]†	$2^{N+o(N)}$	$2^{N+o(N)}$	exp.
Hhan [Hha26]†	$2^{N/2+o(N)}$	—	exp.
Our results			
Field-tower modules	$2^{O(N)}$	$2^{O(N)}$	poly.
Fixed-rank modules	$d^{d/(2\mathrm{e})+o(d)}$	$d^{d/(2\mathrm{e})+o(d)}$	poly.
Semistable lattices	—	$N^{N/(2\mathrm{e})+o(N)}$	poly.
Cyclotomic modules	—	$d^{(n-1+1/\mathrm{e})d/2+o(d)}$	poly.
Logarithmic-rank modules	$(\log N)^{N/(2\mathrm{e})+o(N)}$	$(\log N)^{N/2+o(N)}$	poly.

Notation and assumptions. N denotes the lattice dimension, d the field degree, n the module rank ($N = nd$), and ρ the root discriminant. The field-tower results assume bounded root discriminant and a bounded-step subfield chain. The fixed-rank results assume $\rho = d^{o(1)}$ and require no subfield chain. The logarithmic-rank result assumes $n = \Theta(\log N)$ and $\rho = O(1)$, without requiring a subfield chain. Further growing-rank bounds obtained via subfield descent are given in Theorem 6.9 and Corollary 6.10. The cyclotomic-module result assumes fixed n; semistable lattices include cyclotomic ideals. All displayed bounds for our algorithms are deterministic, use polynomial space, and omit polynomial factors in the input length. † Randomized algorithms.

In particular, cyclotomic ideal lattices are semistable under compatible inner products, so these bounds apply without any restriction on their root discriminants. For higher-rank cyclotomic modules of fixed rank n, we further obtain a deterministic CVP algorithm running in

$$d^{((n-1)/2+1/(2\mathrm{e}))d+o(d)}$$

time (Theorem 5.4).

Extensions to growing module rank. While the first two approaches initially focus on fixed-rank modules, we extend both to modules of growing rank. We obtain improved rank dependence, with factors $n^{N/(2\mathrm{e})}$ for SVP and $(n!)^{d/2}$ for CVP, where n is the module rank, d is the field degree, and $N = nd$ (Section 6). The key technical step is to extend the squared-radius allocation and subset-product analysis of Hanrot and Stehlé [HS07] to module flags, including the unreduced intermediate flags arising during construction.

In particular, for modules of rank $n = \Theta(\log N)$ over totally real or CM fields with bounded root discriminant, we obtain deterministic polynomial-space algorithms for exact SVP and CVP in times

$$(\log N)^{N/(2\mathrm{e})+o(N)} \qquad \text{and} \qquad (\log N)^{N/2+o(N)},$$

respectively (Corollary 6.8).

Table 1 summarizes our main algorithmic results and compares them with existing exact SVP and CVP algorithms.

1.2 Technical Overview

Our module algorithms use a common framework: construct a reduced module flag and enumerate candidate vectors through its rank-one blocks. This framework requires two routines on a rank-one module: exact SVP and enumeration in a Euclidean ball with an arbitrary rational center and

rational squared radius. We implement these routines in two ways. Subfield descent views a rank-one module as a module over a smaller field and recurses. The SVP-oracle approach instead reduces ball enumeration to exact SVP in at most one additional dimension, without requiring a subfield chain.

The latter reduction also applies to arbitrary lattices. Combined with a covering-radius bound, it yields our CVP algorithms for semistable lattices directly. Finally, we improve the dependence on module rank by allocating a shared squared-distance budget across the blocks; this refinement applies to both implementations.

1.2.1 Exact Search by Subfield Descent

Module geometry. Let K be a number field of degree $d = [K : \mathbb{Q}]$. A nonzero vector v spans the K-line $Kv = \{\alpha v : \alpha \in K\}$, whose real span has dimension d. For a rank-r module M, a module flag plays the role of Gram–Schmidt decomposition: we successively choose a K-line and project orthogonally onto its complement. The associated blocks $L_1, \ldots, L_r$ are rank-one modules of dimension d. The flag is reduced if, at each step, the chosen line is generated by a shortest nonzero vector of the current projected module. Compatibility of the inner product ensures that the orthogonal projections preserve the module structure.

The same smaller-field routines will construct reduced flags and enumerate lattice points. Their analysis therefore requires geometric bounds that remain valid during flag construction, before reducedness has been established.

For a rank-one module L, let $H = \lambda_1(L)$ and $\rho = \mathrm{rd}(K)$. Then

$$\frac{H}{\sqrt{d}} \le \det(L)^{1/d} \le \sqrt{\frac{\rho}{d}}\, H. \tag{1}$$

The lower bound follows from Minkowski's theorem, whereas the upper bound uses the number-field structure (Lemma 2.1; see also [AS26]). Thus the normalized determinant of a rank-one module is controlled by its minimum and the root discriminant. No analogous upper bound holds for general lattices: an orthogonal basis with lengths $1, T, \ldots, T$ has minimum one but normalized determinant $T^{(d-1)/d}$, which grows without bound.

To see how this inequality controls enumeration, consider a reduced flag with two blocks of minima H_1, H_2. Applying Minkowski's theorem to the full lattice and then using (1) gives $H_1 \le \sqrt{2\rho H_1 H_2}$, hence $H_2 \ge H_1/(2\rho)$. At search radius $R = H_1$, we have $R/H_1 = 1$ and $R/H_2 \le 2\rho$. A packing argument bounds the number of points in a shifted radius-R ball of L_i by $(1+4R/H_i)^d$. Consequently, both block searches have $2^{O(d)}$ candidates when ρ is bounded.

For longer flags, the block minima need not be monotone, so pairwise comparisons are insufficient. We partition blocks with minima below R into maximal consecutive runs, adjoining to each run the preceding block whose minimum is at least R. Minkowski's theorem and (1) then control the product of the ratios R/H_i over each interval. The intervals are disjoint, so their estimates combine without charging a block more than once. During flag construction, the first block need not yet be reduced; lower bounds on prefix determinants replace that missing condition. The resulting product bound (Lemma 2.3) controls the total number of candidates in the searches used by our algorithms.

This is where the module structure matters. Arbitrarily grouping coordinates of a general lattice into blocks provides neither (1) nor an efficient procedure for searching the blocks. Treating an arbitrary N-dimensional lattice as a module over $\mathbb{Q}$ instead gives module rank N, so the rank-dependent cost remains large.

Recursive ball enumeration. The first approach exploits a chain of subfields. Passing from K to a subfield F leaves the lattice and its metric unchanged but changes its module description.

If $[K : F] = s$, a rank-one K-module becomes a rank-s module over F, with rank-one blocks of dimension

$$d_F = [F : \mathbb{Q}] = d/s.$$

Thus one recursion decreases module rank within a fixed field, while the other decreases field degree when a rank-one block is reached. Together they reduce the search to a bounded-dimensional base case.

The blocks form an orthogonal decomposition of the ambient real space, but not necessarily an orthogonal direct sum of lattices. To enumerate a shifted ball, we first fix a point in the last projected block, lift it to the original lattice, and search the preceding blocks with the induced target and remaining squared radius. Thus even an SVP search centered at zero generates shifted-ball queries. The remaining radius can be arbitrarily small relative to a block minimum, making a bound uniform over all radii essential. We represent all targets and squared radii exactly using rational coordinates.

A two-dimensional example. As a simple illustration, view $\mathbb{Z}[\sqrt{2}]$ as a $\mathbb{Z}$-lattice with basis $(1, 1+\sqrt{2})$ and target coordinates $(0, \frac{1}{2})$. For coefficients $(a, b) \in \mathbb{Z}^2$, the squared distance in the canonical trace norm is

$$2(a + b - \tfrac{1}{2})^2 + 4(b - \tfrac{1}{2})^2.$$

At squared radius $3/2$, the second term restricts b to 0 or 1. For $b = 0$, the remaining target for a is $1/2$, giving $a \in \{0, 1\}$; for $b = 1$, it is $-1/2$, giving $a \in \{-1, 0\}$. All four points lie on the boundary of the ball. The example illustrates how fixing a later coordinate induces a shifted target and a residual radius for the earlier coordinate.

To see how the recursion controls cost, consider a quadratic step $[K : F] = 2$. A rank-one K-module L of dimension $D = [K : \mathbb{Q}]$ becomes a rank-two module over F, with each block having dimension

$$d_F = [F : \mathbb{Q}] = D/2.$$

Once a reduced flag with blocks L_1, L_2 has been prepared, we enumerate L_2 and, for each projected point, exhaust the induced shifted query in L_1.

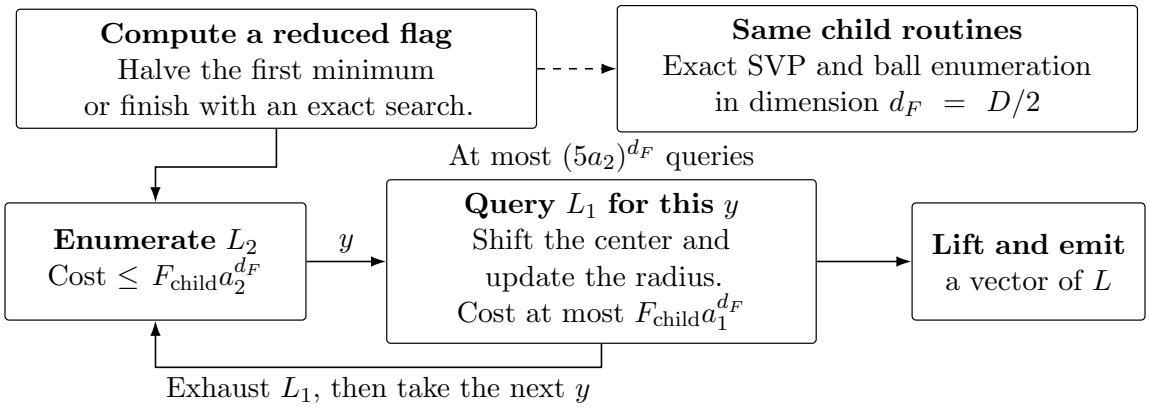
Write $H_i = \lambda_1(L_i)$, $a_i = \max\{1, R/H_i\}$, and $A = \max\{1, R/\lambda_1(L)\}$, where R is the parent radius. Suppose a child search in block i costs at most $F_{\text{child}} a_i^{d_F}$, up to polynomial input-length factors. Packing gives at most $(5a_2)^{d_F}$ points in L_2. We therefore pay for one search in L_2 and at most $(5a_2)^{d_F}$ searches in L_1, rather than multiplying two child-search costs. The product bound of Lemma 2.3 gives $a_1 a_2 \le CA^2$, where C depends only on the field parameters.

For a general step $[K : F] = s$, the child dimension is $d_F = [F : \mathbb{Q}] = D/s$, and the analogous estimate is

$$\prod_{i=1}^{s} \max\{1, R/H_i\} \le C_s A^s, \qquad (C_s A^s)^{d_F} = C_s^{d_F} A^D, \tag{2}$$

where $C_s = 2^{s+1} s^{s/2} \rho^{s/2}$ suffices for root discriminant at most ρ (Lemma 2.3). The key point is that the exponent of A remains the parent dimension D, allowing the estimate to propagate down the subfield chain without additional dimension-dependent exponents.

Constructing the reduced flag. The remaining issue is to construct a reduced flag rather than assume one is given. We start with the first vector of an LLL-reduced basis and recursively prepare the projected suffix. Let H denote the minimum of the current first block, computed by the smaller-field routine. Minkowski's theorem provides computable upper bounds on the minima of the prefixes. If all these bounds exceed $H/2$, an exact search at radius H satisfies the product



$$\begin{aligned} F_{\text{child}}\left[a_2^{d_F} + (5a_2)^{d_F} a_1^{d_F}\right] &\le 2 \cdot 5^{d_F} F_{\text{child}} (a_1 a_2)^{d_F} \\ &\le 2(5C)^{d_F} F_{\text{child}} A^{2d_F} \\ &= 2^{O(D)} F_{\text{child}} A^D. \end{aligned}$$

Figure 1: One quadratic descent step. The displayed sum bounds the cost of the L_2 search and the induced L_1 queries in the prepared flag. Preparation is included in Theorem 3.1. Factors polynomial in the input length are omitted.

bound and finds a shortest vector. We take the K-line it spans as the new first line and prepare the projected suffix.

Otherwise, some prefix contains a nonzero vector of length at most $H/2$, but searching it directly may be too expensive. We instead scale the first block by a dyadic factor $0 < a \le 1$ and search at radius $R = aH/2$. A prefix bound involving $j \ge 2$ blocks scales as $a^{1/j}$, while the radius scales as a. Hence the ratio between the prefix bound and the search radius increases as a decreases.

Choose the smallest dyadic scale at which some prefix bound is still at most R. At the next smaller scale, every prefix bound exceeds the new search radius $R/2$. Thus, at the selected scale, all prefix bounds exceed $R/2$ but at least one is at most R. The corresponding prefix contains a nonzero vector within the search radius and satisfies the product bound (Lemma 2.4). Undoing the scaling gives a vector of original length at most $H/2$, whose K-line replaces the current first line.

Each unsuccessful preparation pass at least halves the first-block minimum. The initial LLL bound limits the number of passes to $O(N)$, while bounded module rank controls the cost of preparing projected suffixes. Importantly, the scaled searches are implemented with the original block metrics and adjusted radii; the smaller-field routines never need to handle a rescaled module metric.

To summarize the recurrence, let $D_j = [K_j : \mathbb{Q}]$ and let F_j be the radius-independent cost factor at field level j, including flag preparation. We obtain $F_j \le 2^{O(D_j)} F_{j+1}$. Since each degree decreases by at least a factor of two, $\sum_j D_j \le 2D_0$. The terminal degree is bounded, so $F_0 = 2^{O(D_0)}$. With bounded module rank, this gives the claimed single-exponential running time for both preparation and exact search.

Closest vectors. CVP requires an additional idea: a ball guaranteed to contain a closest vector can be large compared with the minimum of the original lattice. We handle this by splitting along a shortest dual vector. Under the compatibility assumption, the dual M^* is again a module.

For a rank-r module M, choose a shortest nonzero $v^* \in M^*$ and set $h = \|v^*\|$. Banaszczyk's

transference theorem [Ban93] gives $\mu(M) \le R = 2rd/h$. Projection onto Kv^* gives a rank-one module Λ with $\Lambda^* = M^* \cap Kv^*$. Applying (1) to this dual module yields

$$\lambda_1(\Lambda) \ge \frac{d}{\rho h}, \qquad \frac{R}{\lambda_1(\Lambda)} \le 2r\rho.$$

The dual minimum h cancels, and the projected search has only $2^{O(d)}$ candidates for fixed rank and bounded ρ.

Each projected candidate induces a shifted CVP instance in a module of rank $r-1$. Since the squared distances of the orthogonal components add, minimizing over these instances recovers the exact closest distance. A second depth-first traversal then streams all closest vectors without storing the full output.

Space complexity. All enumerations proceed depth-first, storing only the current recursive branch and the states needed to resume suspended searches. Intermediate data have polynomial encoding length by Lemma 3.3. If S_j denotes the workspace at field level j and P bounds local polynomial storage, then

$$S_j \le \nu_j S_{j+1} + P, \qquad \nu_j = D_j/D_{j+1}.$$

Because $\prod_j \nu_j \le N$, this recurrence gives polynomial workspace. The fixed-rank CVP recursion preserves this bound.

Recovering field arithmetic and the subfield chain. Our complexity bounds include recovering the field arithmetic and a suitable subfield chain; neither is assumed to be provided. For an ideal specified by a defining polynomial and a rational ideal basis, multiplication by a field generator θ is represented by a rational matrix A_θ. An element of K acts by an integral matrix on the ideal precisely when it belongs to $\mathcal{O}_K$. One implication uses ideal stability, and the other follows from the characteristic polynomial of the integral action matrix. Thus integer linear algebra in $\mathbb{Q}[A_\theta]$ recovers the maximal-order arithmetic. The involution can likewise be recovered using adjoints with respect to the Gram matrix (Corollary E.2).

For chain recovery, the algorithm of [vKN13] produces at most d principal subfields, including all maximal proper subfields. Any suitable chain can be refined through maximal proper subfields without increasing the bound on the relative degrees. Searching these candidates yields a tree of depth $O(\log d)$ with at most $d^{O(\log d)}$ nodes. We represent candidate subfields in the original field coordinates and traverse the tree depth-first, retaining polynomial space (Lemma 3.6). Since $d \le N$, this preprocessing fits within the single-exponential time bound.

1.2.2 Ball Enumeration with an SVP Oracle

The second approach implements ball enumeration using an exact-SVP oracle, without any subfield-chain assumption. Given an m-dimensional lattice $L = B_L\mathbb{Z}^m$, we enumerate $(t+L) \cap B(0,R)$. The key is to construct a full-rank sublattice $\Gamma \subseteq L$ satisfying $\lambda_1(\Gamma) > 2R$. Distinct points in the same coset of Γ would be more than $2R$ apart, so each coset contains at most one point of the target ball.

We construct such a sublattice by testing index-p sublattices of the form

$$\Gamma_a = \{B_L z : z_1 + az_2 + \cdots + a^{m-1}z_m \equiv 0 \pmod p\}.$$

For a suitable prime $p = O(mU)$, where $U = |L \cap B(0,2R)|$, some choice of a excludes every nonzero vector of $L \cap B(0,2R)$ from Γ_a. An exact-SVP call certifies the separation condition, and an SVP call on an augmented lattice of dimension $m+1$ identifies the candidate in each coset, if one exists.

The resulting reduction makes $O(mU)$ SVP-oracle calls, uses polynomial working space, and does not require advance knowledge of U (Lemma 4.2). Importantly, the reduction itself imposes

no restriction on the root discriminant. For rank-one module blocks, packing and Gaussian-mass estimates bound U using the block minimum, determinant, and root discriminant. Together with the reduced-flag inequalities, these bounds give $2^{O(d)}\rho^{O(d)}$ oracle calls at fixed module rank. When $\rho = d^{o(1)}$, this overhead is $d^{o(d)}$, yielding deterministic exact SVP and CVP in time $d^{d/(2e)+o(d)}$ (Theorem 4.1).

1.2.3 Semistability and Cyclotomic CVP

The SVP-oracle reduction also applies beyond module lattices when a geometric property bounds the number of points in the search ball. Let L be a D-dimensional semistable lattice: every nonzero sublattice has normalized determinant at least $g = \det(L)^{1/D}$. The covering-radius bound of [LRR26] gives $\mu(L) < 2\sqrt{D}\, g$.

Choose a computable approximation $\widetilde{g}$ with $g \le \widetilde{g} < 2g$ and set $R = 2\sqrt{D}\,\widetilde{g}$. Every target then has a closest vector within distance R. The Voronoi cells have volume g^D and are contained in balls of radius $\mu(L)$ around their centers. Hence a volume comparison gives

$$|L \cap B(0, 2R)| \le \frac{\operatorname{vol} B(0, 2R + \mu(L))}{g^D} = 2^{O(D)}.$$

The SVP-oracle reduction therefore solves exact CVP with $2^{O(D)}$ oracle calls, giving deterministic time $D^{D/(2e)+o(D)}$ and randomized time $D^{D/(4e)+o(D)}$, with success probability at least $2/3$ in the randomized case (Theorem 5.2 and Corollary 5.3).

For a cyclotomic field $K = \mathbb{Q}(\zeta_m)$, multiplication by ζ_m is an irreducible rational isometry of each fractional ideal equipped with a compatible inner product. Grayson's criterion [Gra84, CN18] therefore implies that these ideal lattices are semistable, regardless of their root discriminants. At higher module rank the full lattice need not be semistable, but the rank-one projection arising from dual splitting is. Applying the semistable enumeration bound to this projection yields deterministic CVP time $d^{((n-1)/2+1/(2e))d+o(d)}$ for fixed rank n (Theorem 5.4).

1.2.4 Growing Module Rank

When the module rank grows, searching every block at the full radius produces an unnecessarily large candidate count. Following the squared-radius allocation idea of Hanrot and Stehlé [HS07], we distribute the total squared-distance budget among the blocks. For a rank-r search with radius R, squared radii are allocated in units of R^2/r. There are at most 4^r allocations, and each block is searched only within its assigned radius.

The cost is then governed by $\prod_i \max\{1, R/(\sqrt{r}\, H_i)\}^d$, so only blocks with small minima contribute nontrivially. We adapt the interval-recombination analysis of [HS07] to the module inequality (1), including the unreduced first block during preparation (Lemma 6.5). The inequality $k\log(m/k) \le m/\mathrm{e}$ for k selected blocks among m yields the improved SVP rank factor $n^{N/(2e)}$ (Theorem 6.2).

For CVP, we use one bound on the total squared distance, starting from a block with largest minimum. Each recursive step removes a suffix, so every block is removed exactly once. The accumulated rank-dependent factor is at most $\mathrm{e}^{N/2}(n!)^{d/2}$ (Theorem 6.7).

For module rank $n = \Theta(\log N)$ and bounded root discriminant, the SVP-oracle approach gives deterministic polynomial-space running times $(\log N)^{N/(2e)+o(N)}$ for SVP and $(\log N)^{N/2+o(N)}$ for CVP (Corollary 6.8); no subfield chain is required. Under the additional field assumptions of the subfield-descent result, the recursive block routines can instead replace the SVP oracle, yielding growing-rank guarantees for the first approach as well (Theorem 6.9 and Corollary 6.10).

1.3 Related Work

Peikert and Rosen [PR07] use bounded root discriminant to obtain an $O(\sqrt{\log N})$ approximation factor in worst-case-to-average-case reductions for ideal-SVP, for a number field fixed in advance. In contrast, our algorithms apply to families of number fields of growing degree, where bounded root discriminant is used to control the number of candidates encountered during exact enumeration.

The algebraic structure of number fields has also been exploited in approximation algorithms for ideal-SVP. Cramer, Ducas, and Wesolowski [CDW17] give a quantum polynomial-time approximation algorithm for ideal-SVP over cyclotomic fields of prime-power conductor, under GRH and additional number-theoretic assumptions, with approximation factor $\exp(\widetilde{O}(\sqrt{d}))$. Pellet-Mary, Hanrot, and Stehlé [PMHS19] study tradeoffs between running time and approximation factors using field-dependent preprocessing under heuristic assumptions.

Module-LLL and module-BKZ algorithms exploit module structure for approximate lattice reduction [LPMSW19, MS20, BKLN26, DEdP25, dPT26]. Kirchner, Espitau, and Fouque [KEF19, KEF20] combine recursion in module rank with descent in field degree for approximate reduction.

In contrast, our work develops exact algorithms for SVP and CVP. The minimum–determinant relation for rank-one modules is a key geometric ingredient in our analysis (Lemma 2.1; see also [AS26]). Our main technical contribution is to control the cost of exact enumeration throughout the construction of reduced module flags, including intermediate flags whose first block is not yet reduced. The resulting enumeration bounds hold uniformly over rational shifts and search radii, enabling both recursive subfield descent and an implementation using exact-SVP oracles.

Tian, Sun, and Zhu [TSZ14] propose a local–global reduction for exact ideal-SVP and CVP over totally real fields. However, a key identity underlying their reduction fails even for $\mathbb{Z}[\sqrt{2}]$, as demonstrated in Appendix C. Consequently, their argument does not establish the claimed exact reduction in the stated generality.

Dadush and Kun [DK16, Theorem 1.2] develop convex-body enumeration using an exact Euclidean CVP oracle. Their deterministic construction of a sparse sublattice relies on repeated access to an enumeration of a finite set [DK16, Section 5.1]. Related sparsification-based reductions between approximate CVP and SVP were developed by Aggarwal et al. [ACK+21]. In contrast, our reduction deterministically enumerates all lattice points in an arbitrary shifted Euclidean ball using only an exact-SVP oracle, both to certify the separation of a suitable sublattice and to recover the unique candidate in each of its cosets, without requiring an a priori bound on the number of lattice points (Lemma 4.2).

1.4 Organization of the Paper

The remainder of this paper is organized as follows. Section 2 develops the common framework of reduced module flags and lattice-point enumeration, together with the necessary preliminaries and input model. Section 3 presents our deterministic single-exponential-time algorithms for exact SVP and CVP using recursive subfield descent. Section 4 develops the alternative SVP-oracle approach, which does not require a subfield chain. Section 5 applies this approach to exact CVP on semistable lattices, including cyclotomic ideal lattices, and extends the analysis to higher-rank cyclotomic modules. Section 6 extends both approaches to modules of growing rank and establishes improved rank-dependent bounds. The appendices provide additional proofs and technical details.

2 Module Geometry and Reduced Flags

This section develops the geometric and algorithmic framework underlying our exact SVP and CVP algorithms on module lattices. Our main tool is a reduced module flag, the module analogue of an HKZ-reduced lattice basis, in which each projected rank-one block contains a shortest vector of the remaining projected module. We establish the geometric bounds needed to control enumeration and describe how such flags are constructed.

A lattice $L = B_L\mathbb{Z}^m$ consists of the integer combinations of the linearly independent columns of B_L. Its determinant is the volume of the fundamental parallelepiped spanned by these columns, measured in their real span. The determinant of the Gram matrix is the square of this volume. Let $\lambda_1(L)$ denote the length of a shortest nonzero vector of L, also called the minimum of L, and $\mu(L) = \sup_{t\in\mathrm{span}_{\mathbb{R}}(L)} \mathrm{dist}(t, L)$ the covering radius. The Euclidean dual is $L^* = \{y \in \mathrm{span}_{\mathbb{R}}(L) : \langle x, y\rangle \in \mathbb{Z} \text{ for all } x \in L\}$.

We use $B_m(t, R)$, or $B(t, R)$, for a closed Euclidean ball, and $B_m(R)$ when its center is zero. For a subspace W, π_W is the orthogonal projection onto W and $W^\perp$ the orthogonal complement. The upright $\mathrm{e} = \exp(1)$ is Euler's number. Unless a base is displayed, log is natural. Throughout, n is the module rank, d the field degree, $N = nd$ the lattice dimension, and $\rho = \mathrm{rd}(K) = |\Delta_K|^{1/d}$ the root discriminant of K, where Δ_K is the field discriminant.

2.1 Lattice Problems and Preliminaries

Let $L = B_L\mathbb{Z}^m$ be an m-dimensional Euclidean lattice, where the columns of B_L are linearly independent. We consider the following exact computational problems.

Shortest Vector Problem (SVP). Given a basis B_L of L, find a nonzero vector $v \in L$ satisfying

$$\|v\| = \lambda_1(L) = \min_{x\in\mathbb{Z}^m\setminus\{0\}} \|B_L x\|.$$

Closest Vector Problem (CVP). Given a basis B_L of L and a target vector $u \in \mathrm{span}_{\mathbb{R}}(L)$, find a vector $v \in L$ satisfying

$$\|v - u\| = \mathrm{dist}(u, L) = \min_{x\in\mathbb{Z}^m} \|B_L x - u\|.$$

In our input model, a lattice is represented by the positive definite rational Gram matrix $G = B_L^T B_L$. A lattice vector $B_L x$ is represented by its integer coordinate vector $x \in \mathbb{Z}^m$. For CVP, the target is given by a rational coordinate vector $t \in \mathbb{Q}^m$, representing $u = B_L t$. Thus, the squared norm of $B_L x$ and its squared distance to the target are

$$\|B_L x\|^2 = x^T G x, \qquad \|B_L x - u\|^2 = (x - t)^T G (x - t).$$

By a slight abuse of notation, we identify a target with its coordinate vector when no confusion arises. In particular, projections and Euclidean distances involving t refer to the corresponding vector $B_L t$.

Let B denote the bit length of the lattice input. We use $\ell(t)$, $\ell(R^2)$, and $\ell(t, R^2)$ for the bit lengths of a target, a squared radius, and their combined encoding, respectively. Unless stated otherwise, all running-time bounds include preprocessing, and all algorithms use polynomial working space (Lemmas 3.3 and 4.8). A write-only output stream is excluded from working space, allowing all closest vectors to be enumerated without storing the entire list.

We use three standard lattice bounds. The LLL algorithm [LLL82] with Lovász parameter 3/4 returns, in polynomial time, a nonzero $v \in L$ satisfying $\|v\| \le 2^{(m-1)/2}\lambda_1(L)$. Minkowski's theorem gives $\lambda_1(L) \le \sqrt{m}\det(L)^{1/m}$, and Banaszczyk's transference theorem [Ban93] gives

$$\mu(L)\lambda_1(L^*) \le 2m. \tag{3}$$

2.2 Module Input and Exact Projections

We now specialize the lattice input model to modules over totally real or CM number fields. The compatibility of the inner product with the field action ensures that duals and orthogonal projections preserve the module structure, allowing SVP and CVP to be reduced to lower-rank modules.

Number fields and module input. Let K be a number field of degree $d = [K : \mathbb{Q}]$. An embedding of K is a field homomorphism $\sigma : K \to \mathbb{C}$. The field K is *totally real* if every embedding has image in $\mathbb{R}$, and is a *CM field* if it is a totally imaginary quadratic extension of a totally real field. For either type of field, there is an automorphism $\star$ satisfying $\sigma(\alpha^\star) = \overline{\sigma(\alpha)}$ for every $\alpha \in K$ and every embedding σ. It is the identity for totally real fields and complex conjugation for CM fields. For example, $\mathbb{Q}(\sqrt{2})$ is totally real, while $\mathbb{Q}(i)$ is CM.

The ring of integers $\mathcal{O}_K$ consists of the elements of K satisfying monic polynomials over $\mathbb{Z}$. The field trace is $\mathrm{Tr}_{K/\mathbb{Q}}(x) = \sum_\sigma \sigma(x)$, and the field discriminant Δ_K is the determinant of the trace form $(x, y) \mapsto \mathrm{Tr}_{K/\mathbb{Q}}(xy)$ on an integral basis of $\mathcal{O}_K$. The canonical squared norm on K is $\sum_\sigma |\sigma(x)|^2$, extended to K^n by summing over coordinates. Under this norm, $\mathcal{O}_K$ has covolume $|\Delta_K|^{1/2}$. A *fractional ideal* is a full-rank $\mathbb{Z}$-lattice in K closed under multiplication by $\mathcal{O}_K$.

A rank-n $\mathcal{O}_K$-module M has underlying lattice dimension $N = nd$. We represent M by a $\mathbb{Z}$-basis, a positive definite rational Gram matrix G, and the integer matrices $A(\alpha)$ describing multiplication by the elements α of a supplied integral basis of $\mathcal{O}_K$. We require the compatibility condition

$$A(\alpha)^T G = G A(\alpha^\star),$$

so that multiplication by $\alpha^\star$ is the adjoint of multiplication by α. The field data and action matrices are included in the input bit length B. Module-SVP and module-CVP refer to the ordinary exact lattice problems on M with this additional structure; CVP allows any target with rational coordinates in the lattice basis.

Compatible metrics and projections. For every nonzero $v \in M$, compatibility implies that there are positive weights w_σ such that

$$\|xv\|^2 = \sum_\sigma w_\sigma |\sigma(x)|^2 \qquad \text{for all } x \in K. \tag{4}$$

The derivation is given in Appendix B.1.

Let V be a K-subspace of $M \otimes_{\mathbb{Z}} \mathbb{Q} \cong K^n$. Since V is invariant under the field action and its adjoint, $V^\perp$ is also invariant, and orthogonal projection onto $V^\perp$ commutes with the field action. Consequently, both $M \cap V$ and $\pi_{V^\perp} M$ are modules with compatible rational Gram matrices.

These modules can be represented exactly in the same input model. An adapted $\mathbb{Z}$-basis of M, whose initial vectors form a basis of $M \cap V$, can be computed using integer kernels and unimodular completion [KB79]. The remaining basis vectors project to a basis of $\pi_{V^\perp} M$. A *lift* of a projected point is a point of M having that projection.

For nonzero $v \in M$, we use the full intersection

$$M \cap Kv = Iv, \qquad I = \{x \in K : xv \in M\},$$

where I is a fractional ideal that may strictly contain $\mathcal{O}_K$. Exact Gram computations and an example distinguishing $M \cap Kv$ from $\mathcal{O}_K v$ are given in Appendix B.2.

Shifted targets. Let W be a K-subspace and let t be a target vector, specified by rational coordinates. Set $L = M \cap W^{\perp}$ and $\Lambda = \pi_W M$. For each $y \in \Lambda$, choose a lift $\widetilde{y} \in M$ and define $t_y = \pi_{W^{\perp}}(t - \widetilde{y})$. The lattice vectors projecting to y are exactly $\widetilde{y} + z$ for $z \in L$. By orthogonality,

$$\|\widetilde{y} + z - t\|^2 = \|y - \pi_W t\|^2 + \|z - t_y\|^2. \tag{5}$$

Thus, each projected point y induces a CVP instance in L with shifted target t_y. If $R \geq \mu(M)$, every closest vector projects into $\Lambda \cap B(\pi_W t, R)$. Enumerating these projected points and minimizing the sum in (5) therefore solves CVP. A second pass can enumerate all closest vectors without storing the entire list.

Dual-vector splitting. To obtain a computable search radius, let M have module rank r. Its Euclidean dual M^* is also an $\mathcal{O}_K$-module of rank r, since the adjoint of multiplication by α is multiplication by $\alpha^\star$. Let v^* be a shortest nonzero vector of M^* and set $h = \|v^*\|$. Define

$$W = Kv^*, \quad \Delta = M^* \cap W, \quad \Lambda = \pi_W M, \quad L = M \cap W^{\perp}, \quad R = 2rd/h. \tag{6}$$

For $x \in M$ and $y \in W$, we have $\langle x, y \rangle = \langle \pi_W x, y \rangle$. Hence

$$\Lambda^* = M^* \cap W = \Delta, \qquad \Lambda = \Delta^*.$$

Since M has lattice dimension rd, Banaszczyk's transference bound (3) gives $\mu(M) \leq 2rd/h = R$.

Finally, let $\delta_1, \ldots, \delta_d$ be a $\mathbb{Z}$-basis of Δ. The equations $\langle x, \delta_i \rangle = 0$ for $1 \leq i \leq d$ impose d independent rational linear constraints, whose common solutions in M form L. Thus L has $\mathbb{Z}$-rank $(r-1)d$ and module rank $r-1$, while Λ has module rank one.

2.3 Rank-One Modules and Reduced Flags

We first establish a relation between the minimum and determinant of a rank-one module. This relation will be used to control the geometry of successive projected blocks.

Lemma 2.1. *Let L be a rank-one module over K, with minimum $H = \lambda_1(L)$ and normalized determinant $G_L = \det(L)^{1/d}$. Then $G_L \leq \sqrt{\frac{\rho}{d}}\, H$. Moreover, L^* is also a rank-one module, and $\lambda_1(L)\lambda_1(L^*) \geq \frac{d}{\rho}$, $\mu(L) \leq 2\rho\lambda_1(L)$.*

Proof. Let $v \in L$ be a shortest nonzero vector. Since L has rank one, we can write $L = Iv$, where

$$I = \{x \in K : xv \in L\}$$

is a fractional ideal. Since $v \in L$ and L is an $\mathcal{O}_K$-module, we have $\mathcal{O}_K \subseteq I$, and hence $\mathrm{Nm}(I) \leq 1$, where $\mathrm{Nm}(I)$ denotes the ratio of the canonical covolume of I to that of $\mathcal{O}_K$.

By (4), there are positive weights w_σ such that $\|xv\|^2 = \sum_\sigma w_\sigma |\sigma(x)|^2$. For CM fields, conjugate embeddings have equal weights. Therefore,

$$\det(L) = \sqrt{|\Delta_K|}\, \mathrm{Nm}(I) \prod_\sigma w_\sigma^{1/2}, \qquad H^d \geq d^{d/2} \prod_\sigma w_\sigma^{1/2},$$

where the second inequality follows from the arithmetic–geometric mean inequality applied to $H^2 = \sum_\sigma w_\sigma$. Using $\mathrm{Nm}(I) \leq 1$ and $|\Delta_K|^{1/d} = \rho$, we obtain $G_L \leq \sqrt{\rho/d}\, H$.

As shown in Section 2.2, the dual L^* is also an $\mathcal{O}_K$-module. Since it spans the same d-dimensional space as L, it has rank one. Its normalized determinant is G_L^{-1}. Applying the first inequality to both L and L^* gives $\lambda_1(L)\lambda_1(L^*) \geq d/\rho$. The transference bound (3) then yields

$$\mu(L) \leq \frac{2d}{\lambda_1(L^*)} \leq 2\rho\lambda_1(L).$$

□

Reduced module flags. We introduce a module analogue of the successive shortest-vector condition in Hermite–Korkine–Zolotarev (HKZ) reduction. Instead of individual basis vectors, we consider rank-one module blocks, each containing a shortest vector of the remaining projected module.

Definition 2.2 (Reduced module flag)**.** *Let M be a rank-n module over K. A* module flag *of M is a chain of K-subspaces*

$$0 = V_0 \subset V_1 \subset \cdots \subset V_n = M \otimes_{\mathbb{Z}} \mathbb{Q}, \qquad \dim_K V_i = i.$$

For $1 \leq i \leq n$, the ith prefix *and* block *are defined by*

$$M_i = M \cap V_i, \qquad L_i = \pi_{V_{i-1}^{\perp}} M_i.$$

The flag is reduced *at index i if*

$$\lambda_1(L_i) = \lambda_1\big(\pi_{V_{i-1}^{\perp}} M\big).$$

It is reduced *if this holds for every $1 \leq i \leq n$.*

When $K = \mathbb{Q}$, the blocks are one-dimensional, and the definition recovers the successive shortest-vector condition of HKZ reduction. No size-reduction condition on an adapted basis is imposed.

Since $\dim_K V_i/V_{i-1} = 1$, each block L_i is a rank-one module over K. We write $H_i = \lambda_1(L_i)$ and $G_i = \det(L_i)^{1/d}$ for its minimum and normalized determinant, respectively. The projected module $\pi_{V_1^{\perp}} M$ has rank $n-1$ and inherits the flag

$$\pi_{V_1^{\perp}} V_2 \subset \cdots \subset \pi_{V_1^{\perp}} V_n,$$

whose blocks are $L_2, \ldots, L_n$. We call this the *projected suffix*. An *adapted basis* of M is a $\mathbb{Z}$-basis whose first id vectors form a basis of M_i for every i. Thus, each block L_i corresponds to d basis vectors. Fixing the coefficients of the later blocks determines shifted targets for the earlier blocks through (5).

Geometric bound for reduced flags. Reducedness controls the minima of consecutive blocks. Suppose that the flag is reduced at index i, and let $j \geq i$ and $k = j - i + 1$. The projected prefix $\pi_{V_{i-1}^{\perp}} M_j$ has dimension kd and determinant $\prod_{u=i}^{j} G_u^d$. Since it is a sublattice of $\pi_{V_{i-1}^{\perp}} M$, its minimum is at least H_i. By Minkowski's theorem and Lemma 2.1,

$$H_i \leq \sqrt{kd} \left(\prod_{u=i}^{j} G_u \right)^{1/k} \leq \sqrt{k\rho} \left(\prod_{u=i}^{j} H_u \right)^{1/k}. \tag{7}$$

2.4 Preparing a Reduced Flag

We describe an algorithm for constructing a reduced module flag using rank-one SVP and lattice-point enumeration as subroutines. The main challenge is to ensure that the searches performed during flag construction have controlled enumeration costs. We address this by rescaling the first block when necessary. The subroutines are implemented by subfield descent in Section 3 and by an exact-SVP oracle in Section 4.

Let M have module rank $r \geq 2$, with a flag whose projected suffix is reduced. Write $H = H_1$ for the first block minimum. For $1 \leq j \leq r$, define

$$R_j = \sqrt{jd} \left(\prod_{i=1}^{j} G_i \right)^{1/j}. \tag{8}$$

Since the prefix M_j has dimension jd and determinant $\prod_{i=1}^{j} G_i^d$, Minkowski's theorem gives $\lambda_1(M_j) \leq R_j$. In particular, if $R_j \leq H/2$, then M_j contains a nonzero vector of length at most $H/2$.

Algorithm 1 starts with an LLL-reduced vector and recursively prepares the projected suffix. If $R_j > H/2$ for every $j \geq 2$, a complete search at radius H finds a shortest vector of M. Otherwise, the first block is rescaled by a dyadic factor $a \in (0, 1]$, and a suitable prefix is searched at radius $aH/2$. The resulting vector has original length at most $H/2$ and replaces the current first vector. The choice of a and the guarantees for these searches are established below.

Algorithm 1: Computing a shortest vector and a reduced module flag

Input : Rank-r module M over K
Output : A reduced module flag and a shortest nonzero vector

if $r = 1$ **then**
  Compute a shortest vector w using the rank-one SVP routine, and return the one-block flag and w
Let v be the first vector of an LLL-reduced basis of M
while *true* **do**
  Let $L_1 = M \cap Kv$, and prepare the projected suffix recursively
  Compute $H = \lambda_1(L_1)$ using the rank-one SVP routine and the prefix bounds R_j from (8)
  if $R_j > H/2$ *for all* $2 \leq j \leq r$ **then**
    Enumerate $M \cap B(0, H)$ and choose a shortest nonzero vector w
    Make Kw the first line, prepare its projected suffix recursively, and return the flag and w
  Choose a dyadic factor a and a prefix as in Lemma 2.4
  Scale the first block by a and search the chosen prefix at radius $aH/2$
  Let v' be a nonzero vector found by the search, expressed in the original coordinates, and set $v = v'$

A run of Algorithm 1 on M is called a *preparation* of M. We call a search on a rescaled prefix a *halving search* and the search at radius H a *final search*. Initially, the LLL bound gives $H \leq 2^{(rd-1)/2} \lambda_1(M)$. Each halving search decreases H by at least a factor of two, while $H \geq \lambda_1(M)$ always. Thus, there are at most $(rd-1)/2$ halving searches and at most $rd+3$ passes through the loop.

Search conditions. A flag of rank $r \geq 2$ satisfies the *search conditions at radius* $R > 0$ if its projected suffix is reduced, $R \leq H_1 \leq 2R$, every prefix bound R_j satisfies $R_j \geq R/2$, and the module contains a nonzero vector of length at most R.

Scaling lengths in the first block by $0 < a \le 1$ changes its minimum from H to aH and the prefix bounds to $R_j(a) = a^{1/j} R_j$. The projected suffix and the underlying block decomposition remain unchanged. At radius $R = aH/2$, the scaled first block has minimum $2R$. The corresponding Gram matrix is given in Appendix B.3.

Under the search conditions, the product of the blockwise ratios $\max\{1, R/H_i\}$ is bounded in terms of r and ρ. This product controls the enumeration costs in Sections 3 and 4.

Lemma 2.3. *Let a flag of rank $r \ge 2$ satisfy the search conditions at radius R. Then*

$$\prod_{i=1}^{r} \max\{1, R/H_i\} \le 2^{r+1} r^{r/2} \rho^{r/2}. \tag{9}$$

Proof. We bound the contribution of each maximal consecutive run of indices j with $H_j < R$. Since $H_1 \ge R$, every such run is preceded by an index i with $H_i \ge R$. Adjoining this index gives an interval $\{i, \dots, i+k-1\}$ whose remaining indices satisfy $H_j < R$. These intervals are disjoint and have total length at most r.

First suppose $i \ge 2$. Since the projected suffix is reduced, (7) gives

$$\prod_{j=i+1}^{i+k-1} \frac{R}{H_j} \le (k\rho)^{k/2} \left(\frac{R}{H_i}\right)^{k-1} \le (k\rho)^{k/2},$$

where the last inequality uses $H_i \ge R$.

If $i = 1$, we instead use the prefix condition $R_k \ge R/2$. By Minkowski's bound and Lemma 2.1, $R_k \le \sqrt{k\rho}(\prod_{j=1}^{k} H_j)^{1/k}$. Consequently,

$$\prod_{j=2}^{k} \frac{R}{H_j} \le 2^k (k\rho)^{k/2} \frac{H_1}{R} \le 2^{k+1} (k\rho)^{k/2},$$

where we used $H_1 \le 2R$.

Finally, multiplying these bounds over the disjoint intervals, whose lengths sum to at most r, gives (9). □

Lemma 2.4. *Let a flag of rank $r \ge 2$ have a reduced projected suffix, and let $H = H_1$. If $R_j > H/2$ for every $j \ge 2$, the flag satisfies the search conditions at radius $R = H$, and $M \cap B(0, H)$ contains a shortest vector of M. Otherwise, one can compute a dyadic factor $a = 2^{-q} \in (0, 1]$ such that*

$$\min_{j \ge 2} R_j(a) \le aH/2, \qquad R_j(a) > aH/4 \quad \text{for every } j \ge 2. \tag{10}$$

With the first block scaled by a, a prefix attaining the minimum in (10) satisfies the search conditions at radius $R = aH/2$. Searching this prefix finds a nonzero vector whose length in the original metric is at most $H/2$.

Proof idea. In the first case, the search conditions hold at radius H, and the first block contains a nonzero vector of length H. Thus, a complete search at this radius finds a shortest vector of M.

Otherwise, choose the smallest dyadic $a = 2^{-q}$ for which $R_j(a) \le aH/2$ for some $j \ge 2$. At the next smaller scale $a/2$, this inequality fails for every $j \ge 2$. Since $R_j(a) = 2^{1/j} R_j(a/2)$, it follows that $R_j(a) > aH/4$ for all $j \ge 2$.

A prefix attaining the minimum in (10) contains, by Minkowski's theorem, a nonzero vector of scaled length at most $aH/2$. The prefix conditions and the scaled first minimum aH ensure that the search conditions hold at this radius. Undoing the scaling gives a vector of original length at most $H/2$. The full proof and exact rational comparisons are given in Appendix B.3. □

3 Exact Algorithms by Subfield Descent

We implement the two rank-one subroutines required by Algorithm 1: exact SVP and enumeration of lattice points in Euclidean balls. The key idea is to restrict scalars along a chain of subfields. A rank-one module over $\mathcal{O}_{K_j}$ becomes a module of rank $[K_j : K_{j+1}]$ over $\mathcal{O}_{K_{j+1}}$, allowing Algorithm 1 to reduce both problems to rank-one blocks over a smaller field. When the relative degrees and root discriminants are bounded, the resulting recursion has single-exponential complexity.

3.1 Rank-One SVP and Ball Enumeration

Fix integers $s_0 \geq 2$ and $d_0 \geq 1$, and a rational constant $C_0 \geq 1$. Suppose we are given a chain of totally real or CM fields

$$K_0 \supsetneq K_1 \supsetneq \cdots \supsetneq K_h, \qquad [K_j : K_{j+1}] \leq s_0, \quad [K_h : \mathbb{Q}] \leq d_0, \quad \mathrm{rd}(K_0) \leq C_0,$$

together with integral bases, multiplication tables, inclusions, and involutions $\star$ compatible with these inclusions. The chain data are included in the input length B. We show in Section 3.4 how to recover such a chain when only its existence is assumed.

Write $D_j = [K_j : \mathbb{Q}]$ and $\nu_j = [K_j : K_{j+1}]$. Since $D_j = \nu_j D_{j+1} \geq 2D_{j+1}$, we have $\sum_{k \geq j} D_k \leq 2D_j$. The root discriminant does not increase when passing to a subfield. Indeed, for $E \subseteq F$,

$$|\Delta_F| = |\Delta_E|^{[F:E]} \mathrm{Nm}_{E/\mathbb{Q}}(\mathfrak{d}_{F/E}), \tag{11}$$

where $\mathfrak{d}_{F/E}$ is the relative discriminant ideal, whose norm is a positive integer. Thus every field in the chain has root discriminant at most C_0. In particular, the bound in (9) holds uniformly at every level, with $C_r = 2^{r+1} r^{r/2} C_0^{r/2}$.

For each $0 \leq j \leq h$, we construct two routines on rank-one $\mathcal{O}_{K_j}$-modules:

- $\mathsf{SV}_j(L)$ returns a shortest nonzero vector of L.
- $\mathsf{EN}_j(L, c, \varrho)$ streams all points of $L \cap B(c, \varrho)$, where c has rational coordinates and $\varrho \geq 0$ has rational square.

For a rank-one module L, set $A(L, \varrho) = \max\{1, \varrho/\lambda_1(L)\}$. The following theorem gives the complexity of these routines.

Theorem 3.1. *The routines SV_j and EN_j can be implemented with factors $F_j = 2^{O(D_j)}$, depending only on s_0, d_0, C_0, such that $\mathsf{SV}_j(L)$ takes time $F_j \,\mathrm{poly}(B, N)$ and $\mathsf{EN}_j(L, c, \varrho)$ takes time*

$$F_j A(L, \varrho)^{D_j} \,\mathrm{poly}(B, N, \ell(c, \varrho^2)).$$

Both routines use polynomial working space. The polynomial factors and space bounds are uniform over all field levels (Lemma 3.3). Moreover, $F_h = O(1)$ and $F_j \leq 2^{O(D_j)} F_{j+1}$ for $j < h$.

We describe the recursive implementation of these routines and prove Theorem 3.1 in Section 3.2. The resulting routines are then used to solve general module SVP and CVP in Section 3.3.

3.2 Rank-One Algorithms via Subfield Descent

We prove Theorem 3.1 by implementing SV_j and EN_j recursively along the given subfield chain. At each level, a rank-one $\mathcal{O}_{K_j}$-module is viewed as a rank-ν_j module over $\mathcal{O}_{K_{j+1}}$. We then apply Algorithm 1, using the routines over K_{j+1} to solve SVP and enumerate lattice points in the rank-one blocks. The enumeration proceeds from the last block to the first, as in Fincke–Pohst enumeration. We first bound the number of block queries.

Lemma 3.2. *Let F be a number field of degree d, and let M be an $\mathcal{O}_F$-module equipped with a rank-r flag and an adapted basis. Write $L_1, \dots, L_r$ for its blocks and $H_i = \lambda_1(L_i)$. Let $t \in \operatorname{span}_{\mathbb{R}} M$ have rational coordinates, and let $R \geq 0$ have rational square.*

Suppose that, for each $i \geq 2$, an oracle streams $L_i \cap B(c, \varrho)$ for every rational center c and radius $\varrho \leq R$ with rational square. For L_1, suppose a routine handles every query arising in the backward search, possibly at a radius exceeding R. Then all points of $M \cap B(t, R)$ can be streamed using at most

$$\sum_{i=1}^{r} \prod_{k>i} \left(1 + \frac{4R}{H_k}\right)^d$$

block-oracle invocations, with polynomial additional work per invocation and per output point. A depth-first implementation stores at most one active or suspended oracle state per block.

Proof. We enumerate block coordinates from last to first. At each step, (5) determines the shifted center and remaining squared radius for the next block. Branches with negative remaining squared radius are discarded, and radius-zero queries are handled by exact membership tests. Since the adapted coordinates are unique, every lattice point is output exactly once.

By a standard packing bound, a ball of radius R contains at most $(1 + 4R/H_k)^d$ points of L_k. Thus, at most $\prod_{k>i}(1 + 4R/H_k)^d$ branches reach block i. Summing over i gives the claimed number of oracle invocations.

Since the bound depends only on the later blocks, it remains valid even when the first block is queried at a radius larger than R. The space bound follows from depth-first traversal. □

We implement SV_j and EN_j recursively, starting from the bottom of the subfield chain. At $j = h$, both routines use direct lattice enumeration in dimension $D_h \leq d_0$. A radius-zero enumeration query simply tests whether its center belongs to the lattice.

For $j < h$, a rank-one $\mathcal{O}_{K_j}$-module L has rank ν_j over $\mathcal{O}_{K_{j+1}}$. To implement $\mathsf{SV}_j(L)$, we apply Algorithm 1 over K_{j+1}, using SV_{j+1} for rank-one SVP and EN_{j+1} for the block searches of Lemma 3.2. The algorithm returns a shortest nonzero vector of L. Similarly, $\mathsf{EN}_j(L, c, \varrho)$ first prepares a reduced flag over K_{j+1} using these subroutines, then enumerates $L \cap B(c, \varrho)$ by the backward search of Lemma 3.2.

One detail concerns the halving searches in Algorithm 1, which temporarily scale the first block by $a \leq 1$. We perform the corresponding block queries in the original metric, using

$$\{x \in L_1 : a^2\|x - c\|^2 \leq q\} = L_1 \cap B(c, \sqrt{q}/a).$$

Thus, the lower-level routines always operate on modules with their original inner products.

Correctness follows by induction on the field levels, using the correctness of Algorithm 1 and Lemma 3.2. Within each field level, recursive preparation of projected suffixes decreases the module rank, so the recursion is well-founded.

Lemma 3.3. *The routines SV_j and EN_j can be implemented in polynomial working space. Moreover, every intermediate number has bit length bounded by a fixed polynomial in B and N, and additionally in $\ell(c, \varrho^2)$ for $\mathsf{EN}_j(L, c, \varrho)$.*

Proof. The uniform bit-length bounds are established in Appendix B.4. Both routines use depth-first enumeration, suspending each active enumeration state while processing a recursive query.

At level j, at most ν_j child states are simultaneously active or suspended. Let S_j denote the workspace at level j, and let P be a uniform polynomial bound on local storage. Then

$$S_j \leq \nu_j S_{j+1} + P.$$

Since $\prod_{k=j}^{h-1} \nu_k = D_j/D_h \le N$, the recurrence gives polynomial working space. All enumeration outputs are streamed. □

Having established the recursive construction and its polynomial-space bound, we now prove the running-time guarantees of Theorem 3.1. The key step is to bound the cost of the lower-level rank-one queries arising from flag preparation and ball enumeration.

Proof of Theorem 3.1. We proceed by induction on $h - j$. At the bottom level $j = h$, let $m = D_h \le d_0$ and take an LLL-reduced basis $b_1, \dots, b_m$ of L. Its Gram–Schmidt lengths satisfy $\|b_i^*\| \ge 2^{-(i-1)/2}\lambda_1(L)$. Thus, coordinate enumeration in a ball of radius ϱ visits at most

$$m \prod_{i=1}^{m} \left(1 + \frac{2\varrho}{\|b_i^*\|}\right) \le O_{d_0}(1) A(L, \varrho)^m$$

nodes. For SVP, the first LLL vector provides an initial radius at most $2^{(m-1)/2}\lambda_1(L)$. Hence $F_h = O(1)$.

Now suppose $j < h$, and write $s = \nu_j$ and $d = D_{j+1}$, so that $D_j = sd$. Assume the claimed bounds hold at level $j + 1$ with factor F_{j+1}.

We first analyze a search performed during the preparation of a rank-r flag, where $r \le s$. Let R be the search radius and set $a_i = \max\{1, R/H_i\}$, with the minima measured in the metric of the search. By Lemmas 2.4 and 2.3, $\prod_i a_i \le C_r$.

By Lemma 3.2, at most $\prod_{k>i}(5a_k)^d$ queries reach block i. Each query costs at most $F_{j+1}a_i^d$, up to a uniform polynomial factor. For the first block in a halving search, the query is performed in the original metric at radius at most $H/2$, where H is its unscaled minimum. Its cost factor is therefore one, which agrees with $a_1 = 1$ in the scaled metric. Consequently, each preparation search costs at most

$$F_{j+1} \sum_{i=1}^{r} a_i^d \prod_{k>i} (5a_k)^d \le r 5^{rd} C_r^d F_{j+1},$$

up to a uniform polynomial factor. Since $r \le s_0$ and C_0, s_0 are fixed, this is $2^{O(D_j)}F_{j+1}$.

Each pass of Algorithm 1 recursively prepares a rank-$(r-1)$ projected suffix, computes the first-block minimum using SV_{j+1}, and performs one search. The final pass requires one additional suffix preparation. By Section 2.4, there are at most $rd + 3$ passes.

Let T_r denote the preparation cost, suppressing uniform polynomial factors. The preceding bounds give

$$T_r \le O(D_j)\big(T_{r-1} + 2^{O(D_j)}F_{j+1}\big), \qquad T_1 = F_{j+1}.$$

Since $r \le s \le s_0$, we obtain $T_s = 2^{O(D_j)}F_{j+1}$.

We next analyze $\mathsf{EN}_j(L, c, \varrho)$. Prepare a reduced flag of L over K_{j+1}. Its first-block minimum is $H_1 = \lambda_1(L)$. Every prefix contains a shortest vector of L, so its Minkowski bound is at least H_1. The flag therefore satisfies the search conditions at radius H_1, and Lemma 2.3 gives $\prod_i \max\{1, H_1/H_i\} \le C_s$.

Setting $A = A(L, \varrho)$, we have $\max\{1, \varrho/H_i\} \le A \max\{1, H_1/H_i\}$, and hence

$$\prod_{i=1}^{s} \max\{1, \varrho/H_i\} \le C_s A^s.$$

The backward-search bound now gives an enumeration cost of at most $s5^{D_j}C_s^d F_{j+1}A^{D_j}$, up to a uniform polynomial factor. Together with the preparation cost, this establishes $F_j \le 2^{O(D_j)}F_{j+1}$.

By Lemma 3.3, all intermediate representations have bit length polynomially bounded in the original input and query lengths. The local arithmetic work is charged to recursive invocations and streamed points, whose numbers are already included in the preceding bounds. Thus the accumulated arithmetic overhead is polynomial.

Finally, since $F_h = O(1)$ and $\sum_{k \geq j} D_k \leq 2D_j$, the recurrence yields $\log F_j = O(D_j)$. The polynomial-space bound follows from Lemma 3.3. □

3.3 Exact SVP and CVP

We now extend the rank-one algorithms of Theorem 3.1 to modules of bounded rank. For SVP, we restrict scalars to the first proper subfield and apply Algorithm 1. For CVP, we use the dual-vector splitting of (6) to reduce the module rank recursively.

Theorem 3.4. *Fix integers $n_0, d_0 \geq 1$, $s_0 \geq 2$, and a rational constant $C_0 \geq 1$. Suppose the input supplies a chain of totally real or CM fields*

$$K_0 \supsetneq \cdots \supsetneq K_h, \qquad [K_j : K_{j+1}] \leq s_0, \quad [K_h : \mathbb{Q}] \leq d_0, \quad \mathrm{rd}(K_0) \leq C_0,$$

together with integral bases, multiplication tables, inclusions, and involutions $\star$ compatible with the inclusions.

For any nonzero $\mathcal{O}_{K_0}$-module M of rank at most n_0 in the input model of Section 2.2, exact SVP can be solved deterministically in time $2^{O(N)} \operatorname{poly}(B, N)$ and polynomial space, where $N = \operatorname{rank}_{\mathbb{Z}} M$ and B includes the chain data. The implied constants depend only on n_0, s_0, d_0, C_0.

Proof. If $h = 0$, direct enumeration suffices since $N \leq n_0 d_0$. Otherwise, regard M as a module over K_1 of rank $r = n[K_0 : K_1] \leq n_0 s_0$. Apply Algorithm 1, using SV_1 for rank-one SVP and the backward search of Lemma 3.2 with EN_1 for ball enumeration. By Lemma 2.4, the algorithm returns a shortest nonzero vector of M.

Since r is bounded, the preparation analysis in the proof of Theorem 3.1 applies with rank bound $n_0 s_0$ and gives time $2^{O(N)} F_1 \operatorname{poly}(B, N)$. As $F_1 = 2^{O(D_1)}$ and $D_1 \leq N$, this is $2^{O(N)} \operatorname{poly}(B, N)$. Polynomial space follows from Lemma 3.3. □

We next solve CVP by enumerating the possible projections of a closest vector onto a rank-one module. The dual-vector splitting ensures that only exponentially many projections need to be considered, while each remaining CVP instance has smaller module rank.

Theorem 3.5. *Under the hypotheses of Theorem 3.4, exact CVP on M for any rational target t, including enumeration of all closest vectors, can be solved deterministically in time $2^{O(N)} \operatorname{poly}(B, N, \ell(t))$ and polynomial space. The implied constants depend only on n_0, s_0, d_0, C_0.*

Proof. We induct on the module rank, with rank zero as the trivial case. Let M_r have rank $r \geq 1$ and target t_r. Compute a shortest nonzero vector $v^* \in M_r^*$ using Theorem 3.4. The dual module has Gram matrix G^{-1} and action matrices $A(\alpha^\star)^T$, all of polynomial encoding length.

Let $h^* = \|v^*\|$ and use the dual splitting W, Δ, Λ, L, R of (6). By (3), $R = 2rD_0/h^* \geq \mu(M_r)$. Since $v^* \in \Delta \subseteq M_r^*$, we have $\lambda_1(\Delta) = h^*$. Moreover, $\Lambda^* = \Delta$, so Lemma 2.1 gives

$$\lambda_1(\Lambda) \geq \frac{D_0}{\mathrm{rd}(K_0) h^*}, \qquad A(\Lambda, R) \leq 2n_0 C_0.$$

Enumerate $y \in \Lambda \cap B(\pi_W t_r, R)$ using EN_0. By packing, there are at most $Z_r = (1 + 8n_0 C_0)^{D_0}$ such points. For each y, choose a lift $\widetilde{y} \in M_r$ and recursively solve CVP in L at the shifted target $t_y = \pi_{W^\perp}(t_r - \widetilde{y})$.

By (5), the minimum squared distance among vectors projecting to y is

$$\|y - \pi_W t_r\|^2 + \operatorname{dist}(t_y, L)^2.$$

Every closest vector projects into the enumerated ball, so minimizing this quantity over the enumerated points solves CVP. To enumerate all closest vectors, we first compute the minimum distance, then repeat the enumeration and stream the vectors attaining it.

Let S_r denote the dual-SVP cost and set $E_r = F_0(2n_0C_0)^{D_0}\operatorname{poly}(B, N, \ell(t))$. The total cost Q_r satisfies

$$Q_r \le 2(S_r + E_r) + 3Z_rQ_{r-1}.$$

Since $r \le n_0$ is bounded, unfolding the recurrence gives $Q_r \le 2^{O(N)}\operatorname{poly}(B, N, \ell(t))$.

The recursion has depth at most n_0, and all enumeration outputs are streamed. The projections, lifts, and shifted targets are obtained by exact rational linear algebra and have polynomial encoding length by Appendix B.4. Together with Lemma 3.3, this gives polynomial working space. □

The preceding bounds assume a fixed upper bound on the root discriminant. The same algorithms also apply when this bound varies with the field. Their explicit dependence on the root discriminant is analyzed in Appendix D.

3.4 Recovering the Subfield Chain

The algorithms of Theorems 3.4 and 3.5 assume that a suitable subfield chain is supplied. We show that this chain can be recovered efficiently from the integral arithmetic of the field, thereby removing the need to provide it as part of the input.

Throughout this subsection, we assume that an integral basis of $\mathcal{O}_K$ and its arithmetic are given. The case of ideal lattices specified by a defining polynomial and a rational ideal basis, without the integral arithmetic of $\mathcal{O}_K$, is treated in Appendix E.

Lemma 3.6. *Fix integers $s_0, s_1 \ge 2$ and $d_0 \ge 1$. Let K be a totally real or CM field of degree D, given by an integral basis of $\mathcal{O}_K$, its multiplication table, and its involution $\star$, with total input length B. Then the following algorithms exist.*

(i) A deterministic algorithm finds a chain

$$K = K_0 \supsetneq K_1 \supsetneq \cdots \supsetneq K_h$$

with $[K_j : K_{j+1}] \le s_0$ and $[K_h : \mathbb{Q}] \le d_0$, if one exists, and otherwise reports that none exists. It runs in time $D^{O(\log D)}\operatorname{poly}(B, D)$ and polynomial working space.

(ii) All subfields $E \subseteq K$ with $[K : E] \le s_1$ can be streamed, possibly with repetitions, in time $D^{O(\log s_1)}\operatorname{poly}(B, D)$ and polynomial working space.

Every field produced is totally real or CM, has root discriminant at most $\operatorname{rd}(K)$, and is returned with an integral basis, multiplication table, involution, and inclusion into K, all of polynomial encoding length. The involution on each subfield is the restriction of $\star$.

Proof. We use the principal subfields of [vKN13, Theorem 1 and Section 2.1], which form a generating family for the subfield lattice under intersection. A field F of degree d has at most d principal subfields. They can be computed by factoring a defining polynomial over F and taking rational kernels, using deterministic polynomial-time factorization over number fields [Len83, Theorem 4.5].

Every maximal proper subfield E of F is principal. Indeed, express E as an intersection of principal subfields. Each member contains E, and at least one is proper. By maximality, every proper member must equal E.

For (i), we recursively examine the principal subfields E of the current field F satisfying $2 \le [F : E] \le s_0$. The search terminates successfully once $[F : \mathbb{Q}] \le d_0$, and otherwise returns failure if no child succeeds.

To see that this search is complete, suppose a chain satisfying the required bounds exists. Refine each step through maximal proper intermediate subfields. The indices of the refined steps multiply to the original index, so each remains at most s_0. The first step of the refined chain therefore leads to a maximal proper subfield, which is principal. Induction on the field degree shows that the search finds a suitable chain.

Each step decreases the degree by at least a factor of two, so the depth is at most $\log_2 D$. Since each node has at most D children, the search visits at most $2D^{\log_2 D}$ nodes.

For (ii), every subfield E with $[K : E] \le s_1$ can be reached through a chain of maximal proper subfields, each of which is principal. Such a chain has length at most $\log_2 s_1$. Hence a depth-first search through principal subfields to this depth, retaining only fields of index at most s_1 in K, outputs every required subfield and visits at most $2D^{\log_2 s_1}$ nodes. We do not store previously visited fields, so repetitions may occur.

It remains to show that the field data used throughout these searches have uniformly polynomial encoding length.

Represent each subfield in the original coordinates of K. For $E \subseteq F$, we have $\mathcal{O}_E = E \cap \mathcal{O}_F$. Let U_F contain an integral basis of $\mathcal{O}_F$ in these coordinates, and let C_E be the rational constraints defining E relative to this basis. Then $U_F \ker_{\mathbb{Z}}(C_E)$ gives an integral basis of $\mathcal{O}_E$. Its multiplication table and inclusion maps follow from exact linear algebra using the field arithmetic.

The involution $\star$ restricts to every subfield E. Indeed, for $x \in E$ and every embedding σ of K, $\sigma(x^\star) = \overline{\sigma(x)}$ depends only on $\sigma|_E$. Thus every automorphism of a normal closure fixing E also fixes $x^\star$, proving $x^\star \in E$. The restricted involution acts as complex conjugation under every embedding of E, so E is totally real or CM. By (11), $\mathrm{rd}(E) \le \mathrm{rd}(K)$ and $|\Delta_E| \le |\Delta_K|$.

To bound the coefficients of the integral bases, consider the positive definite form

$$h_E(x, y) = \frac{\mathrm{Tr}_{E/\mathbb{Q}}(xy^\star)}{[E : \mathbb{Q}]} = \frac{\mathrm{Tr}_{K/\mathbb{Q}}(xy^\star)}{D}$$

on E. Before using a subfield, we LLL-reduce its integral basis with respect to this form.

Write $m = [E : \mathbb{Q}]$. For every nonzero algebraic integer x, the arithmetic–geometric mean inequality gives $h_E(x, x) \ge 1$. The covolume of $\mathcal{O}_E$ in this metric is $\sqrt{|\Delta_E|}/m^{m/2}$. Minkowski's second theorem, together with the cube contained in the Euclidean unit ball, therefore yields

$$\prod_{i=1}^{m} \lambda_i(\mathcal{O}_E, h_E) \le \sqrt{|\Delta_E|} \le \sqrt{|\Delta_K|}.$$

Since every successive minimum is at least one, an LLL-reduced integral basis has lengths at most $2^m \sqrt{|\Delta_K|}$.

The discriminant of K is computed from its trace form, so $\log |\Delta_K|$ is polynomial in B, D. Moreover, the Gram matrix of h_K has polynomial encoding length and smallest eigenvalue at least $\exp(-\operatorname{poly}(B, D))$. The basis vectors of $\mathcal{O}_E$ have integer coordinates in $\mathcal{O}_K$, and the normalized trace forms agree on E. Their coordinates in the original integral basis are consequently bounded in absolute value by $\exp(\operatorname{poly}(B, D))$.

We also obtain defining polynomials of uniformly polynomial encoding length. Let $\omega_1, \dots, \omega_m$ be an integral basis of E, and consider $\alpha_c = \sum_{i=1}^{m} c^{i-1}\omega_i$ for integers $0 \le c \le m^3$. For two distinct embeddings of E, equality of their values at α_c is a nonzero polynomial equation in c of degree at most $m-1$. Thus fewer than $\binom{m}{2}(m-1) < m^3$ values are excluded, and some α_c generates E over $\mathbb{Q}$. Primitivity is tested by the rational rank of $1, \alpha_c, \dots, \alpha_c^{m-1}$, and its minimal polynomial is recovered by linear algebra.

Every invocation of the principal-subfield algorithm therefore receives data of uniformly polynomial encoding length. Each node requires $\mathrm{poly}(B, D)$ operations, giving the claimed time bounds. The depth-first searches store field data at only $O(\log D)$ active levels and hence use polynomial working space. □

We can now remove the supplied-chain assumption from the main algorithms.

Corollary 3.7. *The conclusions of Theorems 3.4 and 3.5 remain valid when the supplied subfield chain is replaced by the promise that such a chain exists.*

Proof. The module input already includes the integral arithmetic and involution of K. Find a suitable chain once using Lemma 3.6(i) and reuse it throughout the algorithms. The search takes $D^{O(\log D)}\,\mathrm{poly}(B, D)$ time and polynomial space, where $D = [K : \mathbb{Q}] \le N$. This overhead is absorbed into $2^{O(N)}\,\mathrm{poly}(B, N)$. □

4 Implementing Module Search via SVP Oracles

The algorithms of Algorithm 1 rely on two rank-one subroutines: finding a shortest vector and enumerating all lattice points in a ball. In Section 3.2, we implemented these routines recursively along a subfield chain. We now give an alternative implementation using ordinary exact SVP oracles, without requiring such a chain.

Rank-one SVP requires a single oracle call in dimension d. The main challenge is ball enumeration, since an SVP oracle returns only one shortest vector rather than all points within a prescribed radius. We show that enumeration can be reduced to SVP calls in dimension at most $d+1$, with the number of calls controlled by the number of lattice points in a centered ball.

To analyze the resulting algorithms, we combine packing bounds with Gaussian estimates for rank-one modules. This yields improved bounds on the number of oracle calls during reduced-flag preparation and CVP recursion.

We first establish these oracle reductions and their costs uniformly in the module rank. We then instantiate the oracles with known exact SVP algorithms to obtain the following fixed-rank result.

Theorem 4.1 (SVP and CVP at fixed module rank)**.** *Fix $n \ge 1$. Let M be a rank-n module over a totally real or CM field K of degree d, in the input model of Section 2.2. Suppose K ranges over a family of fields with root discriminant $\rho = \mathrm{rd}(K) = d^{o(1)}$.*

Exact SVP, exact CVP for every rational target, and enumeration of all closest vectors can be solved deterministically in polynomial space and time

$$\mathrm{poly}_n(B, d, \ell(t))\, d^{d/(2\mathrm{e})+o(d)}.$$

Here poly_n denotes a polynomial whose degree may depend on the fixed rank n, and $\ell(t) = 0$ for SVP.

The proof is completed in Section 4.5, after establishing the required oracle-call bounds. The same reductions also yield randomized algorithms with improved running time.

4.1 Ball Enumeration via an SVP Oracle

We first implement the ball enumeration required by Algorithm 1 using an exact SVP oracle. The oracle takes a positive definite rational Gram matrix and returns a shortest nonzero vector in coefficient coordinates. Our reduction applies to arbitrary lattices and uses SVP calls in at most one additional dimension. The number of calls is controlled by the number of lattice points in a centered ball of twice the search radius.

The idea is to partition the lattice into cosets of a sublattice whose minimum exceeds twice the search radius. More precisely, if $\Gamma \subseteq L$ satisfies $\lambda_1(\Gamma) > 2R$, then each coset of Γ contains at most one point of $(t + L) \cap B_m(R)$. Otherwise, the difference of two such points would be a nonzero vector of Γ of length at most $2R$.

We construct a suitable sublattice with small index and recover the point in each coset, if one exists, using an SVP oracle. Let $L = B_L\mathbb{Z}^m$. For a prime p and $a \in \mathbb{F}_p$, define the index-p sublattice

$$\Gamma_a = \left\{ B_L x : x \in \mathbb{Z}^m, \quad \sum_{i=1}^{m} a^{i-1} x_i \equiv 0 \pmod{p} \right\}. \tag{12}$$

We test these sublattices using exact SVP until one satisfies $\lambda_1(\Gamma_a) > 2R$, without enumerating the short vectors of L.

Once a suitable $\Gamma = B_L C\mathbb{Z}^m$ is found, let y represent a coset of L/Γ. Consider the augmented lattice

$$\mathcal{E}_{\Gamma,y} = \left\{ \big(B_L C z + k(t + y),\, kR\big) : z \in \mathbb{Z}^m,\ k \in \mathbb{Z} \right\}. \tag{13}$$

An exact SVP call in dimension $m + 1$ determines whether this coset contains a point in the search ball. The augmented Gram matrix is rational because the original Gram matrix, the coordinates of t, and R^2 are rational.

The complete procedure is given in Algorithm 2.

Lemma 4.2. *Let L be an m-dimensional lattice given by a positive definite rational Gram matrix, let t have rational lattice coordinates, and let $R^2 \geq 0$ be rational. Set*

$$U = |L \cap B_m(2R)|.$$

All points of $(t + L) \cap B_m(R)$ can be streamed without repetition using at most $16mU$ exact SVP calls of dimension at most $m + 1$. The algorithm is deterministic, uses polynomial working space, and requires neither U nor the short vectors of L as input.

Proof. The case $R = 0$ requires only an exact membership test, so assume $R > 0$. We first bound the number of calls needed to find a suitable sublattice.

For each prime p, the sublattice Γ_a has index p and an explicit coefficient basis consisting of pe_1 and $e_i - a^{i-1}e_1$ for $2 \leq i \leq m$, with powers reduced modulo p. Thus each candidate can be constructed by polynomial-time arithmetic and tested with one exact SVP call.

Suppose $p \geq mU$. No nonzero vector of pL has length at most $2R$: otherwise, if pv were such a vector, the $2p + 1$ distinct points $0, \pm v, \ldots, \pm pv$ would all lie in $L \cap B_m(2R)$, contradicting $p \geq mU$. Consequently, for each nonzero $B_L x \in L \cap B_m(2R)$, the polynomial

$$f_x(X) = x_1 + x_2 X + \cdots + x_m X^{m-1}$$

is nonzero modulo p and has at most $m - 1$ roots in $\mathbb{F}_p$. Since there are $U - 1$ nonzero short vectors, fewer than $(m - 1)(U - 1) < p$ values of a are excluded. Hence some Γ_a satisfies $\lambda_1(\Gamma_a) > 2R$.

Algorithm 2: Ball enumeration using an exact SVP oracle

Input : Rational Gram matrix of $L = B_L\mathbb{Z}^m$, shift t with rational lattice coordinates, and rational squared radius $R^2 \geq 0$

Output : All points of $(t + L) \cap B_m(R)$, without repetition

if $R = 0$ **then**
  Output 0 if $-t \in L$, and return
Test whether $\lambda_1(L) > 2R$
if *the test succeeds* **then**
  $\Gamma \leftarrow L$
else
  $p \leftarrow 2$, $\Gamma \leftarrow \perp$
  while $\Gamma = \perp$ **do**
    for $a = 0, \ldots, p - 1$ **do**
      Construct Γ_a using (12)
      if $\lambda_1(\Gamma_a) > 2R$ **then**
        $\Gamma \leftarrow \Gamma_a$ and exit the inner loop
    if $\Gamma = \perp$ **then**
      Replace p by the smallest prime at least $2p$
for *each coset representative* y *of* L/Γ **do**
  Call the SVP oracle on $\mathcal{E}_{\Gamma,y}$ in (13)
  if *the last coefficient is* $k = \pm 1$ **then**
    Normalize its sign and recover the corresponding point $x \in t + L$
    if $\|x\| \leq R$ **then**
      Output x

The algorithm tests primes starting from 2, replacing each prime p by the smallest prime at least $2p$. By Bertrand's postulate, the next prime is less than $4p$. Thus the first tested prime reaching mU is less than $4mU$, and the sum of all tested primes is less than twice this value. The sublattice search therefore uses fewer than $8mU$ SVP calls. The primes can be found deterministically in polynomial time using primality testing [AKS04].

We next verify the recovery step. Fix a coset representative y. If $t + y + \Gamma$ contains a point x with $\|x\| \leq R$, then $\mathcal{E}_{\Gamma,y}$ contains (x, R) of norm at most $\sqrt{2}R$. Every nonzero vector of $\mathcal{E}_{\Gamma,y}$ with $k = 0$ has norm greater than $2R$, since $\lambda_1(\Gamma) > 2R$, while every vector with $|k| \geq 2$ has norm at least $2R$. Therefore, whenever the coset meets the ball, every shortest nonzero vector of the augmented lattice has $k = \pm 1$.

After normalizing the sign, the returned vector gives the unique point of $t + y + \Gamma$ in the ball. Conversely, if the returned vector has $|k| \neq 1$ or its recovered point lies outside the ball, the coset contains no point in the ball. Since distinct cosets are disjoint and each contains at most one such point, every required point is output exactly once.

For Γ_a, the cosets are represented by jB_Le_1 for $0 \leq j < p$. The final prime is less than $4mU$, so fewer than $4mU$ augmented SVP calls are needed. Together with the initial test and the sublattice search, the total number of calls is at most

$$1 + 8mU + 4mU \leq 16mU.$$

If the initial test succeeds, taking $\Gamma = L$ requires only two calls.

Finally, the algorithm stores only the current sublattice, one coset representative, one oracle state, and one output point. The candidate indices and all intermediate representations have polynomial encoding length in the input size. Hence the algorithm uses polynomial working space. □

Subtracting one point from the others shows that every ball of radius R contains at most $|L \cap B_m(2R)|$ lattice points. In the next subsection, we bound this quantity for rank-one module blocks.

4.2 Counting Rank-One Block Candidates

The oracle-call bound of Lemma 4.2 depends on the number of lattice points in a centered ball of twice the search radius. We now bound this number for rank-one modules using two complementary estimates: packing in terms of the minimum and Gaussian mass in terms of the determinant. Taking the better of these bounds will allow us to control the total enumeration cost across the blocks of a reduced flag.

For a lattice L of dimension m, define its Gaussian mass by

$$\Theta_s(L - t) = \sum_{x \in L} \mathrm{e}^{-\pi \|x-t\|^2/s^2}.$$

Poisson summation gives

$$\begin{aligned} \Theta_s(L - t) &= \frac{s^m}{\det L} \sum_{y \in L^*} \mathrm{e}^{-\pi s^2 \|y\|^2} \mathrm{e}^{2\pi i \langle y, t \rangle} \\ &\le \Theta_s(L) = \frac{s^m}{\det L} \Theta_{1/s}(L^*). \end{aligned} \tag{14}$$

Lemma 4.3. *Let L be a rank-one module of dimension d, with normalized determinant $G_L = \det(L)^{1/d}$. For any $c \ge 1/(\pi\sqrt{2})$, every ball of radius $R \le c\sqrt{\rho d}\, G_L$ contains at most*

$$8\big(c\sqrt{2\pi \mathrm{e} \rho}\big)^d$$

points of L.

Proof. Set $s = c\sqrt{2\pi\rho}\, G_L$ and $b = 4\pi^2 c^2 \ge 2$. By Lemma 2.1 applied to L^*,

$$\lambda_1(L^*) \ge \frac{\sqrt{d/\rho}}{G_L}.$$

Consequently, every nonzero $v \in L^*$ satisfies

$$\mathrm{e}^{-\pi s^2 \|v\|^2} \le \mathrm{e}^{-(b-1)d/2} \mathrm{e}^{-\pi s^2 \|v\|^2/b}.$$

Poisson summation and monotonicity of Gaussian mass imply $\Theta_{\sqrt{b}/s}(L^*) \le b^{d/2} \Theta_{1/s}(L^*)$. Therefore,

$$\begin{aligned} \Theta_{1/s}(L^*) - 1 &\le \mathrm{e}^{-(b-1)d/2} \Theta_{\sqrt{b}/s}(L^*) \\ &\le (b\mathrm{e}^{1-b})^{d/2} \Theta_{1/s}(L^*). \end{aligned}$$

Since $b \ge 2$, we have $(b\mathrm{e}^{1-b})^{d/2} \le \sqrt{2/\mathrm{e}} < 7/8$, and hence $\Theta_{1/s}(L^*) < 8$.

By (14), it follows that $\Theta_s(L - t) \le 8(s/G_L)^d$. Every point in a ball of radius $R \le c\sqrt{\rho d}\, G_L$ contributes at least $\mathrm{e}^{-d/2}$ to this Gaussian mass. Thus the number of such points is at most $8(\sqrt{\mathrm{e}}\, s/G_L)^d$, as claimed. □

We combine this estimate with the packing bound. Let $H = \lambda_1(L)$ and define the mixed scale

$$u(L,R) = \min\left\{\max\left\{1, \frac{R}{H}\right\}, \max\left\{\sqrt{\rho}, \frac{R}{\sqrt{d}\,G_L}\right\}\right\}. \tag{15}$$

Packing gives

$$|L \cap B_d(2R)| \le (1 + 4R/H)^d \le 5^d \max\{1, R/H\}^d.$$

On the other hand, let $\bar{u} = \max\{\sqrt{\rho}, R/(\sqrt{d}\,G_L)\}$. Since $2R \le 2\bar{u}\sqrt{d}\,G_L$, applying Lemma 4.3 with $c = 2\bar{u}/\sqrt{\rho} \ge 2$ gives

$$|L \cap B_d(2R)| \le 8(2\sqrt{2\pi \mathrm{e}}\,\bar{u})^d \le 8 \cdot 9^d \bar{u}^d.$$

Taking the better of the two bounds yields

$$|L \cap B_d(2R)| \le 8 \cdot 9^d u(L,R)^d. \tag{16}$$

Finally, consider an adapted basis of a rank-r module with blocks $L_1, \ldots, L_r$. In backward enumeration, fixing the later block coordinates determines the center and remaining radius of the next query through (5). Every remaining radius is at most R.

Write $U_i = |L_i \cap B_d(2R)|$. By Lemma 4.2, each query to block L_j uses at most $16dU_j$ SVP-oracle calls. Moreover, at most $\prod_{i>j} U_i$ branches reach that block, since every radius-R ball contains at most U_i points of L_i. The total number of calls is therefore

$$\sum_{j=1}^{r} 16dU_j \prod_{i>j} U_i \le 16dr \prod_{i=1}^{r} U_i. \tag{17}$$

In the next subsection, we bound this product for the flags arising in Algorithm 1.

4.3 Preparing Reduced Module Flags

We now analyze the oracle complexity of Algorithm 1. The main step is to bound the product of the mixed scales (15) for flags satisfying the search conditions of Section 2.4.

Lemma 4.4. *Let $r \ge 2$, and suppose a rank-r flag satisfies the search conditions of Section 2.4 at radius R. Writing $u_i = u(L_i, R)$, we have*

$$\prod_{i=1}^{r} u_i \le 2^{r+1} r^{r/2} \rho^{(r-1)/2}. \tag{18}$$

Consequently, backward enumeration at radius R, using Algorithm 2 for each block, requires at most

$$16dr\left[\Upsilon_r \rho^{(r-1)/2}\right]^d$$

SVP-oracle calls, where $\Upsilon_r = 2 \cdot 144^r r^{r/2}$.

Proof. Write $H_i = \lambda_1(L_i)$ and $G_i = \det(L_i)^{1/d}$, and set

$$\theta = \frac{1}{2}\log\rho, \qquad x_i = \log(H_i/R), \qquad \delta_i = \log\frac{\sqrt{d}\,G_i}{H_i}.$$

Minkowski's bound and Lemma 2.1 give $0 \le \delta_i \le \theta$. The search conditions also give $0 \le x_1 \le \log 2$.

If $x_i \geq 0$, then $\log u_i = 0$. Otherwise, with $t_i = -x_i > 0$, (15) gives

$$\log u_i = \min\{t_i, \max\{\theta, t_i - \delta_i\}\}. \tag{19}$$

Consider a maximal consecutive run of indices with $x_j < 0$. Adjoin its immediately preceding index i, where $x_i \geq 0$, to obtain an interval $\{i, \dots, i+k-1\}$. These intervals are disjoint, and their total length is at most r.

Suppose first that $i \geq 2$. Since the projected suffix is reduced, (7) gives

$$H_i \leq \sqrt{kd} \left(\prod_{j=i}^{i+k-1} G_j \right)^{1/k}.$$

Substituting $G_j = H_j \mathrm{e}^{\delta_j}/\sqrt{d}$ and using $x_i \geq 0$ and $\delta_i \leq \theta$, we obtain

$$\sum_{j=i+1}^{i+k-1} t_j \leq \theta + \sum_{j=i+1}^{i+k-1} \delta_j + \frac{k}{2} \log k.$$

If $i = 1$, the prefix condition $R_k \geq R/2$ instead gives

$$\sum_{j=2}^{k} t_j \leq \theta + \sum_{j=2}^{k} \delta_j + x_1 + k \log 2 + \frac{k}{2} \log k.$$

We use the following elementary observation. If q nonnegative numbers t_j and numbers $0 \leq \delta_j \leq \theta$ satisfy $\sum_j t_j \leq \theta + \sum_j \delta_j + b$, then

$$\sum_j \min\{t_j, \max\{\theta, t_j - \delta_j\}\} \leq q\theta + b.$$

Indeed, the claim is immediate if every summand is at most θ. Otherwise, let J be the nonempty set of summands exceeding θ. On J the summand equals $t_j - \delta_j$, while outside J it is at most t_j. Hence the sum is at most

$$\theta + \sum_{j \notin J} \delta_j + b \leq q\theta + b.$$

Applying this observation to the negative indices in each interval, we obtain a contribution of at most θ per negative index, plus the corresponding interval error. There are at most $r-1$ negative indices, and the total error is bounded by $(r+1)\log 2 + (r/2)\log r$, because the interval lengths sum to at most r and $x_1 \leq \log 2$. Therefore

$$\sum_{i=1}^{r} \log u_i \leq \frac{r-1}{2} \log \rho + (r+1) \log 2 + \frac{r}{2} \log r.$$

Exponentiating proves (18).

Finally, let $U_i = |L_i \cap B_d(2R)|$. By (16) and (18),

$$\begin{aligned} \prod_{i=1}^{r} U_i &\leq 8^r 9^{rd} \left(\prod_{i=1}^{r} u_i \right)^d \\ &\leq \left[\Upsilon_r \rho^{(r-1)/2} \right]^d. \end{aligned}$$

The oracle-call bound follows from (17). □

The exponent $(r-1)/2$ in (18) comes from combining the minimum- and determinant-based bounds and is essential for the complexity estimates below.

Theorem 4.5. *For each fixed module rank n, Algorithm 1 computes a reduced module flag and a shortest nonzero vector using at most*

$$\mathrm{poly}_n(d)\, A_n^d \rho^{(n-1)d/2}$$

exact SVP-oracle calls of dimension at most $d+1$, where A_n depends only on n.

Proof. Let S_r denote the number of SVP-oracle calls at rank r. A rank-one instance requires one call, so $S_1 = 1$.

At rank $r \geq 2$, each pass of Algorithm 1 prepares a rank-$(r-1)$ projected suffix, computes the first-block minimum with one SVP call, and performs a backward search. By Lemma 4.4, the search uses at most $16dr[\Upsilon_r \rho^{(r-1)/2}]^d$ calls.

There are at most $rd+3$ passes by Section 2.4, and the final pass requires one additional preparation of the projected suffix. Thus

$$\begin{aligned} S_r &\leq (rd+4) S_{r-1} \\ &\quad + (rd+3)\left(1 + 16dr[\Upsilon_r \rho^{(r-1)/2}]^d\right). \end{aligned} \tag{20}$$

The same bound applies to shorter prefixes, since $\Upsilon_j \rho^{(j-1)/2}$ is nondecreasing in j. Correctness follows from Lemma 2.4.

Unfolding (20) for fixed n gives

$$S_n \leq \mathrm{poly}_n(d)\, \Upsilon_n^d \rho^{(n-1)d/2}.$$

Taking $A_n = \Upsilon_n$ proves the theorem. □

4.4 CVP via Dual Splitting

We next analyze the oracle complexity of CVP using the dual-vector splitting of (6), as in Theorem 3.5. A shortest dual vector determines a rank-one projection, and each projected candidate gives a CVP instance of smaller module rank. We use Gaussian counting to bound the number of candidates.

To enumerate all closest vectors in polynomial space, we use two passes: the first computes the minimum distance, and the second streams the vectors attaining it. If a ball of radius R contains a closest vector, the two passes of Lemma 4.2 require at most $32mU$ SVP-oracle calls, where $U = |L \cap B_m(2R)|$.

We begin with rank-one CVP.

Lemma 4.6. *For a rank-one module L in the input model of Section 2.2 and a rational target, exact CVP and enumeration of all closest vectors require at most*

$$64d(160\sqrt{\rho})^d$$

SVP-oracle calls of dimension at most $d+1$.

Proof. Compute $h = \lambda_1(L^*)$ using a d-dimensional SVP call and set $R = 2d/h$. By (3), every target has a closest lattice point within distance R.

Let $G_L = \det(L)^{1/d}$. Applying Lemma 2.1 to L^* gives

$$\frac{1}{G_L} \le \sqrt{\frac{\rho}{d}}\, h, \qquad \frac{R}{G_L} \le 2\sqrt{\rho d}.$$

Thus $2R \le 4\sqrt{\rho d}\, G_L$, and Lemma 4.3 with $c = 4$ yields

$$U = |L \cap B_d(2R)| \le 8(4\sqrt{2\pi e\rho})^d \le (160\sqrt{\rho})^d.$$

Including the dual-SVP call, the two enumeration passes require at most $1 + 32dU \le 64d(160\sqrt{\rho})^d$ calls. □

For higher rank, let v^* be a shortest nonzero vector of M^*, with $h = \|v^*\|$, and let W, Δ, Λ, L be the splitting of (6). The projected lattice Λ has rank one, while L has rank one less. We first bound the projected candidates.

Lemma 4.7. *Let M have rank $r \ge 2$, and let Λ and $R = 2rd/h$ be as in (6). Every ball of radius R contains at most*

$$Z_r = 8(2r\sqrt{2\pi e\rho})^d \tag{21}$$

points of Λ. These points can be streamed using at most

$$16d(160r\sqrt{\rho})^d$$

SVP-oracle calls of dimension at most $d + 1$.

Proof. Write $G_\Delta = \det(\Delta)^{1/d}$ and $G_\Lambda = \det(\Lambda)^{1/d}$. Since $\Lambda^* = \Delta$, we have $G_\Lambda = G_\Delta^{-1}$. Moreover, $\lambda_1(\Delta) = h$. By Lemma 2.1,

$$\frac{R}{G_\Lambda} = RG_\Delta \le 2r\sqrt{\rho d}.$$

Applying Lemma 4.3 with $c = 2r$ proves (21).

For enumeration, apply the same lemma at radius $2R$ with $c = 4r$:

$$|\Lambda \cap B_d(2R)| \le 8(4r\sqrt{2\pi e\rho})^d \le (160r\sqrt{\rho})^d.$$

The oracle-call bound follows from Lemma 4.2. □

Let C_r denote the number of SVP-oracle calls needed to compute the minimum distance and stream all closest vectors at module rank r. Let S_r be the preparation bound of (20). By Lemma 4.6, $C_1 \le 64d(160\sqrt{\rho})^d$.

For $r \ge 2$, apply Algorithm 1 to M^* to obtain a shortest dual vector, using at most S_r oracle calls. By transference, $R = 2rd/h$ is a valid search radius for every target.

Enumerate $y \in \Lambda \cap B(\pi_W t, R)$. For each y, choose a lift $\widetilde{y} \in M$ and solve CVP recursively in L at $t_y = \pi_{W^\perp}(t - \widetilde{y})$. By (5), the minimum squared distance among vectors projecting to y is

$$\|y - \pi_W t\|^2 + \operatorname{dist}(t_y, L)^2.$$

Minimizing this expression over all projected candidates solves CVP. A second pass revisits the candidates and streams precisely the closest vectors, without storing the candidate list.

There are at most Z_r projected candidates, and each pass uses at most $16d(160r\sqrt{\rho})^d$ oracle calls to enumerate them. Consequently,

$$C_r \le S_r + 32d(160r\sqrt{\rho})^d + 2Z_r C_{r-1}, \qquad r \ge 2. \tag{22}$$

We solve this recurrence together with (20) in the next subsection.

4.5 Uniform Bounds and Algorithmic Consequences

We now solve the SVP and CVP recurrences without fixing the module rank. The resulting oracle-call bounds will also be used in the growing-rank analysis of Section 6. We then instantiate the oracles with known exact SVP algorithms to obtain the fixed-rank running times stated in Theorem 4.1.

We first record the uniform bit-complexity bound needed to convert oracle-call counts into running times.

Lemma 4.8. *The oracle reductions of this section can be implemented in polynomial working space, with every stored number having bit length bounded by a fixed polynomial in B, N, and $\ell(t, R^2)$, whose degree is independent of the module rank.*

The technical proof, which also covers their use in Section 6, is given in Appendix B.4.

Theorem 4.9. *Let the module rank $n \geq 1$ be part of the input, and set $N = nd$ and $P = N + 6$. The algorithms of Section 4 solve exact SVP and CVP, including enumeration of all closest vectors, using at most $\mathcal{S}_{n,d}$ and $\mathcal{C}_{n,d}$ exact SVP-oracle calls of dimension at most $d + 1$, where*

$$\mathcal{S}_{n,d} \leq P^{n+4} 2^{O(N)} n^{N/2} \rho^{(N-d)/2}, \tag{23}$$

$$\mathcal{C}_{n,d} \leq P^{2n+8} 2^{O(N)} (n!)^d \rho^{N/2}. \tag{24}$$

The implied constants are absolute, and both reductions use polynomial working space.

Proof. We first bound the SVP calls. Let $X_r = \Upsilon_r^d \rho^{(r-1)d/2}$ for $r \geq 2$. Since $rd + 4 \leq P$, the recurrence (20) implies

$$S_r \leq P S_{r-1} + 17 N P X_r.$$

As X_r is nondecreasing and $S_1 = 1$, unfolding the recurrence gives

$$S_r \leq P^{r-1} + 17 N P X_r \sum_{j=0}^{r-2} P^j \leq P^{r+4} X_r.$$

The bound (23) follows from $\Upsilon_n = 2 \cdot 144^n n^{n/2}$ and $\Upsilon_n^d \leq 512^N n^{N/2}$. The case $n = 1$ follows directly from $S_1 = 1$.

For CVP, normalize the call count by setting $c_r = C_r / \rho^{rd/2}$. Using $r^{r/2} \leq r!$ and $Z_r \leq 8(10r\sqrt{\rho})^d$, the recurrence (22) yields

$$\begin{aligned} c_1 &\leq 64 d \, 160^d, \\ c_r &\leq 2P^{r+4}(288^r r!)^d + 64 d (160r)^d + 64 (10r)^d c_{r-1}. \end{aligned} \tag{25}$$

We claim that $c_r \leq P^{2r+8} 512^{rd} (r!)^d$. The claim holds for $r = 1$. For $r \geq 2$, substitute the induction hypothesis into (25). Using $P = N + 6$, $r \leq n$, and $r! = r(r-1)!$, each of the three terms is at most one eighth of the claimed bound. The induction therefore closes, proving (24).

Finally, Lemma 4.8 gives polynomial bounds on intermediate encoding lengths and working space. All searches are performed depth-first, reusing their oracle workspace. □

We now obtain concrete running times by instantiating the exact SVP oracle. By Lemma F.1, an SVP query specified by a rational Gram matrix reduces in polynomial time to an integer-basis SVP instance of the same dimension. Kannan's algorithm [Kan83], with the analysis of Hanrot and Stehlé [HS07, Theorem 2], solves such instances deterministically in polynomial space and time

$$m^{m/(2\mathrm{e})+o(m)}$$

in dimension m, up to polynomial factors in the input length.

Proof of Theorem 4.1. Fix the module rank n and suppose $\rho = d^{o(1)}$. Since $N = nd$, (23) and (24) give

$$\mathcal{S}_{n,d}, \mathcal{C}_{n,d} \le \mathrm{poly}_n(d)\, 2^{O(d)} \rho^{O(d)} = d^{o(d)}.$$

Every oracle call has dimension at most $d+1$ and hence costs

$$(d+1)^{(d+1)/(2\mathrm{e})+o(d)} = d^{d/(2\mathrm{e})+o(d)}$$

up to a polynomial factor in its input length.

By Lemma 4.8, all oracle inputs have uniformly polynomial encoding length, and the reductions use polynomial working space. Multiplying the oracle cost by the number of calls yields

$$\mathrm{poly}_n(B, d, \ell(t))\, d^{d/(2\mathrm{e})+o(d)},$$

as claimed. □

The same reductions admit a faster randomized implementation using the exact SVP algorithm of [FGL26]. Since the oracle queries are adaptive, we apply Lemma F.2 to amplify their success probabilities and guarantee correctness of the complete output.

Corollary 4.10. *Under the hypotheses of Theorem 4.1, there are randomized algorithms for exact SVP and CVP for every rational target, including enumeration of all closest vectors without repetition, that succeed with probability at least* $2/3$ *in time*

$$\mathrm{poly}_n(B, d, \ell(t))\, d^{d/(4\mathrm{e})+o(d)}$$

and polynomial working space on every execution.

Proof. By (23) and (24), together with $n^{n/2} \le n!$, the integer

$$Q = \left\lceil P^{2n+8} 1024^N (n!)^d \rho^N \right\rceil$$

bounds the number of SVP-oracle calls in either reduction. This bound is computable from the input, since $\rho^N = |\Delta_K|^n$. All calls have dimension at most $d+1$.

For fixed n and $\rho = d^{o(1)}$, we have $Q = d^{o(d)}$ and $\log Q = o(d \log d)$. By Lemma F.2, the running time is

$$\begin{aligned} &\mathrm{poly}_n(B, d, \ell(t))\, Q(1+\log Q)(d+1)^{(d+1)/(4\mathrm{e})+o(d)} \\ &\quad = \mathrm{poly}_n(B, d, \ell(t))\, d^{d/(4\mathrm{e})+o(d)}. \end{aligned}$$

The uniform bit bounds of Lemma 4.8 and the sequential oracle repetitions give polynomial working space and the claimed per-execution resource bounds. □

5 Semistability and Cyclotomic CVP

For exact SVP, applying Kannan's algorithm directly to an N-dimensional lattice already gives running time $N^{N/(2\mathrm{e})+o(N)}$. For cyclotomic fields with growing root discriminant, the module-based reduction of Section 4 does not improve the leading exponent of this bound. We therefore focus on CVP, where semistability allows us to obtain faster algorithms without requiring a small root discriminant.

Our approach combines the SVP-oracle-based ball enumeration of Section 4.1 with covering-radius bounds for semistable lattices. We first obtain CVP algorithms for arbitrary semistable lattices and show that cyclotomic ideal lattices satisfy this condition. We then extend the results to higher-rank cyclotomic modules using the dual splitting of Section 4.4.

5.1 Semistability and the Search Radius

Let L be a D-dimensional Euclidean lattice and $g = \det(L)^{1/D}$. The lattice L is *semistable* if every nonzero $\mathbb{Z}$-sublattice $\Gamma \subseteq L$ of rank k satisfies $\det(\Gamma) \geq g^k$. It suffices to consider primitive sublattices, since saturation does not increase the determinant. This is the definition of semistability in [Gra84, CN18].

Equivalently, the normalized lattice L/g is stable: it has determinant one, and every nonzero sublattice has determinant at least one. Its dual gL^* is also stable [RS24, Section 2.4].

The covering theorem of Letwin, Reis and Rothvoss [LRR26, Theorem 1.2] gives

$$\mu(L) \leq \sqrt{\frac{2\mathrm{e}}{\pi}}\,\sqrt{D}\,g < 2\sqrt{D}\,g. \tag{26}$$

Indeed, the theorem applies to L/g because its dual is stable. Applying it to the Euclidean unit ball and using $\mathbb{E}\|Z\| \leq \sqrt{\mathbb{E}\|Z\|^2} = \sqrt{D}$ for $Z \sim N(0, I_D)$ gives the bound above. For $D = 1$, it follows directly from $\mu(L) = g/2$.

We use this covering-radius bound through the following general reduction.

Lemma 5.1. *Let L be a D-dimensional lattice given by a positive definite rational Gram matrix, and let $g = \det(L)^{1/D}$. Suppose a rational $\kappa \geq 1$ satisfies $\mu(L) \leq \kappa\sqrt{D}\,g$. For every rational target t, a closest vector and the list of all closest vectors can be computed deterministically using at most*

$$32D(21\kappa)^D$$

exact SVP-oracle calls of dimension at most $D+1$. The reduction uses polynomial working space apart from the oracle workspace.

Proof. For each $x \in L$, its Voronoi cell has volume $\det(L) = g^D$ and is contained in $B(x, \mu(L))$. The Voronoi cells have disjoint interiors, so the cells centered at points of $L \cap B_D(2u)$ are contained in $B_D(2u + \mu(L))$. Using $\operatorname{vol} B_D(1) \leq (2\pi\mathrm{e}/D)^{D/2}$, we obtain

$$|L \cap B_D(2u)| \leq \left(\sqrt{2\pi\mathrm{e}}\left(\frac{2u}{\sqrt{D}\,g} + \kappa\right)\right)^D \tag{27}$$

for every $u \geq 0$.

We choose a search radius whose square is rational. If G is the input Gram matrix, then $g^{2D} = \det G \in \mathbb{Q}$. By exact comparisons of rational powers, we can compute a positive dyadic rational $\widetilde{g}$ with $g \leq \widetilde{g} < 2g$ in polynomial time. Set $R^2 = \kappa^2 D\widetilde{g}^2$. Then R^2 is rational and $\mu(L) \leq R < 2\kappa\sqrt{D}\,g$. Applying (27) with $u = R$ gives

$$U := |L \cap B_D(2R)| \leq (5\sqrt{2\pi\mathrm{e}}\,\kappa)^D \leq (21\kappa)^D.$$

Since the ball of radius R about t contains a closest vector, one pass of Lemma 4.2 determines the minimum squared distance. A second pass streams exactly the vectors attaining this minimum. Each pass uses at most $16DU$ SVP calls, proving the claimed bound.

The radius and all oracle inputs have polynomial encoding length. Both passes are performed sequentially, storing only one candidate and one oracle workspace at a time. □

5.2 CVP for Semistable Lattices

The preceding reduction gives an exact CVP algorithm for every semistable lattice, without requiring any module structure.

Theorem 5.2. *Let L be a semistable lattice of dimension D given by a positive definite rational Gram matrix, and let t be a rational target. A deterministic reduction finds a closest vector to t, or streams all closest vectors without repetition, using at most $32D\,42^D$ exact SVP-oracle calls of dimension at most $D+1$.*

The reduction uses polynomial working space apart from the oracle workspace. With Kannan's exact SVP algorithm, it runs in deterministic time

$$\mathrm{poly}(B, D, \ell(t))\, D^{D/(2\mathrm{e})+o(D)}$$

and polynomial working space.

Proof. By (26), Lemma 5.1 applies with $\kappa = 2$, giving at most $32D\,42^D$ SVP calls. Kannan's algorithm [Kan83], with the analysis of Hanrot and Stehlé [HS07, Theorem 2], solves exact SVP in dimension m in deterministic time $m^{m/(2\mathrm{e})+o(m)}$, up to polynomial input-length factors, and polynomial space. Since every oracle call has dimension at most $D+1$, the claimed running time follows. □

The randomized exact SVP algorithm of [FGL26] further improves the leading exponent. The adaptive-oracle amplification of Lemma F.2 ensures correctness of the complete output, including enumeration of all closest vectors.

Corollary 5.3. *Under the hypotheses of Theorem 5.2, a randomized algorithm finds a closest vector, or streams all closest vectors without repetition, with probability at least $2/3$ in time*

$$\mathrm{poly}(B, D, \ell(t))\, D^{D/(4\mathrm{e})+o(D)}$$

and polynomial working space on every execution.

Proof. The reduction of Theorem 5.2, including both enumeration passes, makes at most $Q = 32D\,42^D$ SVP calls of dimension at most $D+1$. By Lemma F.2, these calls can be implemented with overall success probability at least $2/3$ and running time

$$\mathrm{poly}(B, D, \ell(t))\, Q(1+\log Q)(D+1)^{(D+1)/(4\mathrm{e})+o(D)}.$$

Since $Q(1+\log Q) = 2^{O(D)}$, this is $D^{D/(4\mathrm{e})+o(D)}$ up to polynomial input-length factors. The resource bounds follow from the same lemma. □

Dependence on the covering bound. The leading running-time exponents in Theorem 5.2 and Corollary 5.3 already follow from the earlier covering-radius bound of Regev and Stephens-Davidowitz [RS24, Theorem 6.2]. Indeed, applying their result to the stable lattice L/g, where $g = \det(L)^{1/D}$, gives

$$\mu(L) \le 4\sqrt{D}(\log D + 10)g.$$

Choosing a computable rational $\kappa = O(\log D)$ in Lemma 5.1 then requires at most

$$2^{O(D \log\log D)} = D^{o(D)}$$

exact-SVP calls, which are absorbed into the $o(D)$ terms of both running-time bounds. The stronger covering bound of [LRR26] allows $\kappa = 2$, reducing the number of calls to $2^{O(D)}$, with the explicit bound $32D\,42^D$ in Theorem 5.2. Both bounds use the same SVP-based ball-enumeration reduction of Lemma 4.2.

5.3 Cyclotomic Ideal Lattices

We now identify a family of semistable lattices arising from algebraic number fields. Grayson's criterion [Gra84, Observation 1.22] (see also [CN18, Theorem 1.5]) states that a lattice admitting an irreducible rational isometry is semistable. Here an isometry is irreducible if it preserves no proper nonzero rational subspace.

Let $K = \mathbb{Q}(\zeta_m)$ be a cyclotomic field, where ζ_m is a primitive mth root of unity, and let $d = [K : \mathbb{Q}]$. On every fractional ideal of K, multiplication by ζ_m has the irreducible mth cyclotomic polynomial as its minimal polynomial. Consequently, this action preserves no proper nonzero rational subspace.

Moreover, $\zeta_m^\star = \zeta_m^{-1}$. By the metric-compatibility condition $A(\alpha)^T G = G A(\alpha^\star)$ of Section 2.2, we have

$$A(\zeta_m)^T G A(\zeta_m) = G.$$

Thus multiplication by ζ_m is an irreducible rational isometry. Grayson's criterion shows that every rank-one module lattice over K, with a compatible inner product, is semistable.

Applying Theorem 5.2 and Corollary 5.3, we obtain deterministic and randomized CVP algorithms for cyclotomic ideal lattices with respective running times

$$\mathrm{poly}(B, d, \ell(t))\, d^{d/(2\mathrm{e})+o(d)} \quad \text{and} \quad \mathrm{poly}(B, d, \ell(t))\, d^{d/(4\mathrm{e})+o(d)}.$$

These bounds do not require the root discriminant to be bounded.

5.4 Higher-Rank Cyclotomic Modules

Although rank-one cyclotomic module lattices are semistable, this property need not hold at higher module rank. Thus Theorem 5.2 cannot be applied directly to an arbitrary higher-rank module.

Instead, we reuse the dual splitting of Section 4.4. The resulting rank-one projection is semistable, allowing us to bound the number of projected candidates without relying on the root discriminant. We then solve the remaining lower-rank CVP instances recursively.

Theorem 5.4. *Fix $n \geq 1$. Let M be a rank-n module over a cyclotomic field K of degree d, in the input model of Section 2.2. For every rational target t, exact CVP and enumeration of all closest vectors can be solved in deterministic time*

$$\mathrm{poly}_n(B, d, \ell(t))\, d^{((n-1)/2+1/(2\mathrm{e}))d+o(d)}$$

and polynomial working space.

Proof. We induct on the module rank. At rank one, the claim follows from Theorem 5.2.

For rank $r \geq 2$, compute a shortest nonzero vector v^* of M^*, with $h = \|v^*\|$, using exact SVP in dimension rd. Let W, Δ, Λ, L be the dual splitting of (6), and set $R = 2rd/h$. By transference, every target admits a closest vector within distance R.

Both Δ and Λ are rank-one modules over the same cyclotomic field and are therefore semistable by Section 5.3. Since $v^* \in \Delta$, semistability applied to $\mathbb{Z}v^*$ gives $G_\Delta \leq h$. Using $G_\Lambda = G_\Delta^{-1}$, we obtain

$$\frac{2R}{\sqrt{d}\, G_\Lambda} = 4r\sqrt{d}\, \frac{G_\Delta}{h} \leq 4r\sqrt{d}.$$

Applying (27) to Λ with $\kappa = 2$ gives

$$U_r := |\Lambda \cap B_d(2R)| \leq \left(\sqrt{2\pi\mathrm{e}}(4r\sqrt{d} + 2)\right)^d.$$

Thus every translated ball of radius R contains at most U_r projected candidates, which can be enumerated using at most $16dU_r$ SVP-oracle calls by Lemma 4.2.

Each projected candidate y determines a CVP instance in $L = M \cap W^{\perp}$ at the shifted target t_y. By (5), the corresponding minimum squared distance is $\|y - \pi_W t\|^2 + \operatorname{dist}(t_y, L)^2$. Taking the minimum over the projected candidates therefore solves CVP.

Let $\mathcal{T}(m)$ denote the deterministic exact-SVP running time in dimension m, suppressing polynomial input-length factors, and let T_r denote the running time for one closest vector at rank r. Define

$$\Omega_r(d) = 32d\big(2^8(1 + 4r\lceil\sqrt{d}\rceil)\big)^d.$$

This bounds both the projected enumeration cost in oracle calls and the number of recursive queries. Hence

$$T_1 \le \operatorname{poly}(d)\, 64^d \mathcal{T}(d+1), \qquad T_r \le \mathcal{T}(rd) + \Omega_r(d)\big(\mathcal{T}(d+1) + T_{r-1}\big). \tag{28}$$

For fixed r, we have $\Omega_r(d) = d^{d/2+o(d)}$. Writing $a = 1/(2\mathrm{e})$, Kannan's SVP bound gives $\mathcal{T}(rd) \le d^{rad+o(d)}$. If b_r denotes the coefficient of $d \log d$ in the resulting upper bound, then

$$b_1 = a, \qquad b_r = \max\{ra, \tfrac{1}{2} + a, \tfrac{1}{2} + b_{r-1}\} = \frac{r-1}{2} + a.$$

The last equality follows by induction from $a < 1/2$. This proves the claimed running time for one closest vector.

To enumerate all closest vectors, apply Lemma F.3. Its additional factor $2^{nd} + 1 = 2^{O(d)}$ for fixed n does not change the leading exponent. All intermediate inputs have polynomial encoding length, and a depth-first traversal stores only one recursive branch. The working space is therefore polynomial. □

Corollary 5.5. *Under the hypotheses of Theorem 5.4, a randomized algorithm solves exact CVP and lists all closest vectors without repetition, with probability at least $2/3$, in time*

$$\operatorname{poly}_n(B, d, \ell(t))\, d^{((n-1)/2+1/(4\mathrm{e}))d+o(d)}$$

and polynomial working space on every execution.

Proof. Let $Q_1 = 32d\,42^d$ and $Q_r = 1 + \Omega_r(d)(1 + Q_{r-1})$ for $r \ge 2$. These computable bounds cover all SVP calls needed to find one closest vector at rank r. By Lemma F.3, the complete enumeration uses at most $Q = (2^{nd} + 1)Q_n$ oracle calls.

All calls have dimension at most $d + 1$, except those computing shortest dual vectors, which have dimension rd at rank r. The number of rank-r dual calls is at most

$$(2^{nd} + 1) \prod_{j=r+1}^{n} (1 + \Omega_j(d)).$$

Applying Lemma F.2, we reduce the failure probability of each adaptive SVP call to at most $1/(3Q)$ using $O(\log Q)$ repetitions. A union bound guarantees that all oracle answers are correct with probability at least $2/3$. For fixed n, we have $\log Q = O_n(d \log(d+2))$ and $\Omega_j(d) = d^{d/2+o(d)}$, so amplification incurs only a polynomial overhead.

The calls of dimension at most $d + 1$ contribute at most

$$d^{((n-1)/2+1/(4\mathrm{e}))d+o(d)}$$

time. The dual calls at rank r contribute at most $d^{((n-r)/2+r/(4e))d+o(d)}$. Since $r \geq 2$ for these calls and $1/(4e) < 1/2$, their exponents are strictly smaller than $(n-1)/2 + 1/(4e)$. Thus the lower-dimensional oracle calls determine the stated running time.

The oracle repetitions are performed sequentially, so the algorithm uses polynomial working space on every execution. □

6 Improved Algorithms at Growing Module Rank

The oracle reductions of Section 4 already apply when the module rank n is part of the input. However, their dependence on n becomes costly as the rank grows: the SVP and CVP bounds contain factors $n^{N/2}$ and $(n!)^d$, respectively, where $N = nd$.

We improve these rank-dependent factors to $n^{N/(2e)}$ for SVP and $(n!)^{d/2}$ for CVP by exploiting a shared bound on the total squared distance across the blocks. For SVP, we combine squared-radius allocation with a Hanrot–Stehlé-type analysis of block minima. For CVP, we use a corresponding enumeration strategy on a prepared module flag. We then derive the consequences at logarithmic module rank and along subfield chains.

Throughout this section, let $N = nd$ and $P = N + 6$.

6.1 Allocating the Squared Radius

The backward searches of Section 4.2 bound each block independently using the full search radius. We instead exploit the fact that the squared distances in orthogonal blocks sum to at most the squared radius.

Following Hanrot–Stehlé [HS07], we discretize the squared distance assigned to each block. For a search of radius R at rank r, a block whose squared contribution lies between $w_j R^2/r$ and $(w_j+1)R^2/r$ is searched with the latter bound. Enumerating these allocations preserves every candidate while reducing the product of the block search radii.

Lemma 6.1. *Let M have a rank-r flag with blocks $L_1, \dots, L_r$ and minima $H_1, \dots, H_r$. Let t be a rational target and $R^2 > 0$ rational. For $w \in \mathbb{Z}^r_{\geq 0}$ with $\sum_j w_j \leq r$, set*

$$r_j(w) = R\sqrt{\frac{w_j+1}{r}}.$$

The points of $M \cap B(t,R)$ can be streamed without repetition using at most

$$16dr \sum_{w} \prod_{j=1}^{r} |L_j \cap B_d(2r_j(w))| \tag{29}$$

SVP-oracle calls of dimension at most $d+1$. Moreover, this number is at most

$$16dr\, 4^r (5\sqrt{2})^{rd} \prod_{j=1}^{r} \max\left\{1, \frac{R}{\sqrt{r}\, H_j}\right\}^d. \tag{30}$$

The sum in (29), without the factor $16dr$, also bounds the number of complete block tuples examined.

Proof. For $x \in M \cap B(t,R)$, let $e_1, \dots, e_r$ be the orthogonal block components of $x - t$. By (5), we have $\|x-t\|^2 = \sum_j \|e_j\|^2$. Assign to x the integers

$$w_j = \left\lfloor \frac{r\|e_j\|^2}{R^2} \right\rfloor.$$

Since $\sum_j w_j \le r$, there are at most $\binom{2r}{r} \le 4^r$ possible allocations.

For each allocation w, traverse the flag backwards and enumerate block j at radius $r_j(w)$, using the center determined by the later coordinates. Retain a candidate only if its exact block squared distances give the prescribed allocation and its total squared distance is at most R^2. The floor rule assigns each point to a unique allocation, including points on the boundary, so the resulting enumeration is complete and has no repetitions.

Let $U_j(w) = |L_j \cap B_d(2r_j(w))|$. By Lemma 4.2, a block query costs at most $16dU_j(w)$ oracle calls. The backward-search analysis of (17) therefore bounds the calls for one allocation by $16dr \prod_j U_j(w)$, proving (29).

For the second bound, packing gives

$$U_j(w) \le 5^d \max\left\{1, \frac{r_j(w)}{H_j}\right\}^d \le 5^d (w_j+1)^{d/2} \max\left\{1, \frac{R}{\sqrt{r}\,H_j}\right\}^d.$$

By the arithmetic–geometric mean inequality, $\prod_j (w_j+1) \le (1 + \sum_j w_j/r)^r \le 2^r$. Thus each allocation contributes at most a factor $2^{rd/2}$ beyond the common product. Summing over at most 4^r allocations proves (30).

All squared radii and floor tests are rational, so the allocation and verification steps are exact. □

6.2 Shortest Vectors via the Hanrot–Stehlé Analysis

When the searches of Algorithm 1 use Lemma 6.1, their cost depends on the product

$$\prod_{j=1}^{r} \max\left\{1, \frac{R}{\sqrt{r}\,H_j}\right\}^d.$$

Only blocks with $H_j < R/\sqrt{r}$ contribute. Unlike the products analyzed in Section 4.3, these blocks need not form consecutive intervals. We therefore adapt the arbitrary-subset analysis of Hanrot and Stehlé [HS07, Theorem 3 and Section 4.3] to the module setting.

The analysis combines consecutive runs of selected indices and applies the interval inequality (7) with weights determined by their relative densities. This controls the selected blocks collectively, rather than paying the interval error separately for each run. The resulting rank dependence comes from maximizing $k\log(m/k)$ over the number k of selected indices.

For $m \ge 1$ and $\rho \ge 1$, define

$$\mathcal{L}_m(\rho) = \max\left\{\log(2\rho), \frac{m\log(m\rho)}{2(m-1)}\right\},$$

where the second term is omitted for $m = 1$, and set

$$\Psi_m(\rho) = m\mathcal{L}_m(\rho) \exp\left(\frac{\log\rho}{2\mathcal{L}_m(\rho)} - 1\right). \tag{31}$$

Equivalently, $\Psi_m(\rho)$ is the maximum of $\frac{k}{2}\log\rho + \mathcal{L}_m(\rho) k\log(m/k)$ over real $0 \le k \le m$, with $k\log(m/k) = 0$ at $k = 0$.

We obtain the following oracle-call bounds.

Theorem 6.2. *Run Algorithm 1 with every search performed by Lemma 6.1. The algorithm remains exact. For module rank $n \ge 2$ given as part of the input, it uses at most*

$$(N+6)^{n+2}\, 2^{O(N)} n^{N/(2\mathrm{e})} \rho^{N/\sqrt{\mathrm{e}}} \tag{32}$$

exact SVP-oracle calls of dimension at most $d+1$. A refined bound is

$$(N+6)^{n+2}\,2^{O(N)}\rho^{d/2}\exp\bigl(d\Psi_{n-1}(\rho)\bigr).$$

The implied constants are absolute.

We first state the bounds on block minima needed for the proof.

Lemma 6.3. *Let $m \ge 1$ and $\rho \ge 1$. For $2 \le k \le m$,*

$$\frac{k\log(k\rho)}{2(k-1)} \le \mathcal{L}_m(\rho).$$

Moreover, $\mathcal{L}_m(\rho)$ is nondecreasing in m, and for $m \ge 2$, $\mathcal{L}_m(\rho) \le \log(\rho\sqrt{2m})$.

The proof is given in Appendix A.1.

Lemma 6.4. *Let $m \ge 2$, and let positive numbers $h_1, \dots, h_m$ satisfy (7), with h in place of H and parameter $\rho \ge 1$, on every interval. For any nonempty $I \subseteq \{1,\dots,m\}$ with $k = |I|$,*

$$\begin{aligned}\frac{h_1^k}{\prod_{i\in I} h_i} &\le (m\rho)^{k/2}\exp\bigl(\mathcal{L}_m(\rho)k\log(m/k)\bigr)\\ &\le (m\rho)^{k/2}(\rho\sqrt{2m})^{k\log(m/k)}.\end{aligned} \tag{33}$$

Consequently,

$$\prod_{i=1}^{m}\max\left\{1,\frac{h_1}{\sqrt{m}\,h_i}\right\} \le (2m)^{m/(2\mathrm{e})}\rho^{m/\sqrt{\mathrm{e}}}. \tag{34}$$

The proof is given in Appendix A.3. The argument follows the recombination method of Hanrot and Stehlé, replacing their HKZ-specific interval estimate by (7). We provide the required recombination argument in Appendix A.2.

During Algorithm 1, the first block need not yet be reduced. The next lemma combines the reduced projected suffix with the prefix-radius conditions of Lemma 2.4. Its bound applies to the intermediate flags encountered during preparation.

Lemma 6.5. *Let a rank-r flag, $r \ge 2$, have a reduced projected suffix. Suppose that $R \le H_1 \le 2R$ and every prefix radius in (8) is at least $R/2$. Set $m = r-1$. For any $I \subseteq \{2,\dots,r\}$ with $k = |I|$,*

$$\begin{aligned}\sum_{j\in I}\log\frac{R}{\sqrt{r}\,H_j} &\le \frac{1}{2}\log(r\rho) + 2\log 2 + \frac{k}{2}\log(4\rho)\\ &\quad + \mathcal{L}_m(\rho)k\log(m/k),\end{aligned} \tag{35}$$

where $k\log(m/k) = 0$ for $k = 0$. Consequently,

$$\begin{aligned}\prod_{j=1}^{r}\max\left\{1,\frac{R}{\sqrt{r}\,H_j}\right\} &\le 4\sqrt{r\rho}\,2^m\exp(\Psi_m(\rho)),\\ \prod_{j=1}^{r}\max\left\{1,\frac{R}{\sqrt{r}\,H_j}\right\} &\le 4\sqrt{r\rho}\,2^m(2m)^{m/(2\mathrm{e})}\rho^{m/\sqrt{\mathrm{e}}}.\end{aligned} \tag{36}$$

The proof is given in Appendix A.4.

Proof of Theorem 6.2. Correctness follows as in Theorem 4.5, since Lemma 6.1 enumerates every point in each searched ball. The halving and final searches therefore retain their original correctness guarantees.

Only the cost of each search changes. At rank r, every searched prefix of rank $r' \le r$ satisfies the search conditions of Lemma 6.5. Combining (30) with the second bound of (36), and bounding the elementary rank-dependent factors, gives a search cost of at most

$$E_r = 16dr\,512^{rd} r^{rd/(2\mathrm{e})} \rho^{rd/\sqrt{\mathrm{e}}}.$$

The rank-one prefix case satisfies the same bound directly. The estimate is uniform over the shorter prefixes, and E_r is nondecreasing in r.

The number of passes and recursive suffix preparations is unchanged. As in (20), the oracle-call count satisfies

$$S_r \le (rd+4)S_{r-1} + (rd+3)(1+E_r) \le PS_{r-1} + 2PE_r.$$

With $S_1 = 1$, unfolding the recurrence gives $S_n \le 5P^{n-1}E_n$. Since $80N \le P^3$, this proves (32), with the factor 512^N absorbed into $2^{O(N)}$.

For the refined bound, use the first estimate in (36) instead. The resulting search bound is nondecreasing in the prefix rank, by Lemma 6.3 and the monotonicity of $u \mapsto u\exp(\log\rho/(2u) - 1)$ for $u \ge \log(2\rho)$. The same recurrence then yields the bound involving $\Psi_{n-1}(\rho)$.

Finally, the allocation procedure introduces only rational squared radii and exact floor tests. The polynomial-space and encoding-length bounds follow from Lemma 4.8. □

6.3 Closest Vectors with a Shared Squared-Distance Bound

The dual-splitting recursion of Section 4.4 removes one module block at a time, leading to a rank-dependent factor $(n!)^d$. We instead adapt the shared-distance enumeration of Kannan [Kan83] and Hanrot–Stehlé [HS07, Section 5] to a prepared module flag. This improves the rank factor to $(n!)^{d/2}$, at the cost of increasing the discriminant factor from $\rho^{N/2}$ to ρ^N.

At each recursive query, we choose a block with maximum minimum among the remaining blocks and enumerate the projection from that block onward under one squared-distance bound. The covering radii control the search radius, while reducedness and Minkowski's theorem control the number of projected candidates. Each candidate leaves a CVP instance on a shorter prefix of the same prepared flag.

Let M have a reduced flag (V_i) as in Section 2.3, with blocks L_i, minima H_i, and normalized determinants G_i. For exact computation of the search radii, choose a positive rational $\widehat{\rho}$ satisfying

$$\rho \le \widehat{\rho} < \frac{5}{4}\rho.$$

Such a number can be computed with polynomially many bits by comparing rational dth powers with $|\Delta_K|$.

Lemma 6.6. *Fix $1 \le r \le n$, and choose $j \in \{1,\dots,r\}$ with $H_j = \max_{i\le r} H_i$. Let $k = r - j + 1$ and set*

$$\overline{R}^2 = 4\widehat{\rho}^2 r H_j^2.$$

Let Λ be the projection of M_r perpendicular to M_{j-1}, with blocks $L_j,\dots,L_r$.

The induced flag of M_r is reduced, every target in its span has a closest point within radius $\overline{R}$, and

$$\frac{H_j^k}{\prod_{i=j}^r \sqrt{d}\,G_i} \le k^{k/2}. \tag{37}$$

For any rational target and rational R^2 with $0 < R^2 \le \overline{R}^2$, the points of Λ within radius R of the projected target can be streamed using at most

$$16N(1024\rho\sqrt{r})^{kd} \tag{38}$$

SVP-oracle calls of dimension at most $d+1$. The number of points output is at most $(1024\rho\sqrt{r})^{kd}$.

Proof. For each $i \le r$, the projection of M_r onto $V_{i-1}^{\perp}$ is contained in the corresponding projection of M and contains L_i. Its minimum is therefore H_i, so the induced flag is reduced.

By Lemma 2.1, each block satisfies $\mu(L_i) \le 2\rho H_i$. Successive rounding in orthogonal blocks consequently gives

$$\mu(M_r)^2 \le \sum_{i=1}^{r} \mu(L_i)^2 \le 4\rho^2 \sum_{i=1}^{r} H_i^2 \le \overline{R}^2.$$

Moreover, reducedness gives $\lambda_1(\Lambda) = H_j$ and $\det(\Lambda) = \prod_{i=j}^{r} G_i^d$. Minkowski's theorem in dimension kd proves (37).

We next bound the projected enumeration. For an allocation w with $\sum_{i=j}^{r} w_i \le k$, let $R_i = R\sqrt{(w_i+1)/k}$ and $\overline{R}_i = \overline{R}\sqrt{(w_i+1)/k}$. Write $a_i = \sqrt{d}\, G_i$. By Lemma 2.1, $a_i \le \sqrt{\rho} H_i \le \sqrt{\rho} H_j$, so

$$\frac{\overline{R}_i}{a_i} \ge \sqrt{\rho}.$$

Applying (16) at radius $R_i \le \overline{R}_i$ gives

$$U_i := |L_i \cap B_d(2R_i)| \le 8 \cdot 9^d \left(\frac{\overline{R}_i}{a_i}\right)^d.$$

Since $\prod_{i=j}^{r}(w_i+1) \le 2^k$, the determinant bound (37) yields

$$\prod_{i=j}^{r} \frac{\overline{R}_i}{a_i} \le (2\sqrt{2}\,\widehat{\rho}\sqrt{r})^k.$$

There are at most 4^k allocations, so

$$\begin{aligned}\sum_{w} \prod_{i=j}^{r} U_i &\le 32^k (18\sqrt{2}\,\widehat{\rho}\sqrt{r})^{kd} \\ &\le (1024\rho\sqrt{r})^{kd},\end{aligned}$$

where the last inequality follows from $d \ge 1$ and $576\sqrt{2} \cdot (5/4) < 1024$.

By (29), multiplying this bound by $16dk \le 16N$ proves (38). The same sum bounds the number of complete projected tuples examined, and hence the number of accepted points. □

The radius $\overline{R}^2$ is rational, since $\widehat{\rho}$ and H_j^2 are rational. It can therefore be used directly in Lemma 6.1, without knowing the closest distance in advance.

Theorem 6.7. *After reduced-flag preparation, Algorithm 3 finds a closest vector. A two-pass variant also streams all closest vectors without repetition. Including preparation, both tasks require at most*

$$P^{3n+6}\, 2^{O(N)} (n!)^{d/2} \rho^N \tag{39}$$

SVP-oracle calls of dimension at most $d+1$, where the implied constant is absolute. The algorithms use polynomial working space apart from the oracle workspace.

Algorithm 3: CVP with a shared squared-distance bound

Input : A prepared rank-r prefix M_r and a rational target t
Output : A closest vector of M_r

if $r = 0$ **then**
  Return the zero vector
Write t in the basis of M_r, let $z = \lfloor t \rfloor$ coordinatewise, and replace t by $t - z$
if $t = 0$ **then**
  Return the lattice vector with coefficient vector z
Choose j maximizing H_i for $i \le r$, and set $k = r - j + 1$
Set $R^2 = 4\widehat{\rho}^2 r H_j^2$
foreach $w \in \mathbb{Z}_{\ge 0}^k$ *with* $\sum_{i=j}^{r} w_i \le k$ **do**
  Enumerate projected tuples using radii $R_i^2 = (w_i + 1)R^2/k$
  foreach *tuple satisfying the allocation tests and total squared-distance bound* **do**
    Form the shifted target in the first $j - 1$ blocks
    Solve CVP recursively in M_{j-1} using the same prepared flag
    Combine the prefix solution with the projected tuple and update the best answer
Add z to the coefficient vector of the best answer and return

Proof. By Lemma 6.6, the search ball contains a closest vector and its projection is among the enumerated candidates. For each projected tuple, (5) reduces the remaining minimization to CVP in the prefix M_{j-1}. Taking the minimum over these candidates therefore solves CVP.

To enumerate all closest vectors, the first pass determines the minimum squared distance $D_{\min}^2$. The second pass revisits the projected candidates and streams the closest prefix vectors whose combined squared distance equals $D_{\min}^2$. Uniqueness of the allocation and the prefix decomposition prevents repetitions.

It remains to bound the oracle calls. Consider a rank-r query whose pivot removes k blocks, and let $X = (1024\rho\sqrt{r})^{kd}$. By Lemma 6.6, one pass uses at most $16NX$ calls for projected enumeration and makes at most X recursive queries. If Q_{r-k} bounds the cost of a recursive prefix query, the two passes together use at most

$$2(16N + 1)X \max\{1, Q_{r-k}\} \le 128NX \max\{1, Q_{r-k}\}$$

oracle calls.

All queries of a given prefix rank use the same pivot, since the prepared flag and its block minima are fixed. Let $r_1 = n, r_2, \dots$ be the successive prefix ranks and $k_i = r_i - r_{i+1}$ the numbers of blocks removed. Since $r_i \le n$ and $\sum_i k_i = n$, we have

$$\prod_i r_i^{k_i} \le \prod_i n^{k_i} = n^{\sum_i k_i} = n^n \le \mathrm{e}^n n!,$$

where the last inequality follows from $\log(n!) \ge \int_0^n \log x \, dx = n \log n - n$.

There are at most n recursive levels. Using $128N \le P^3$, multiplication of the per-level bounds gives

$$Q_n \le P^{3n}(1024\sqrt{\mathrm{e}})^N (n!)^{d/2} \rho^N.$$

The preparation cost (23) is absorbed into (39), since $n^{N/2} \le \mathrm{e}^{N/2}(n!)^{d/2}$ and $\rho \ge 1$. This proves the oracle-call bound.

The enumeration proceeds depth-first, storing only one allocation and one recursive search path at each level. Polynomial bounds on the intermediate encoding lengths follow from Lemma 4.8. □

The bounds of Theorems 4.9 and 6.7 can be compared before processing a target. Thus the CVP algorithm may use whichever bound is smaller for the given parameters.

6.4 Logarithmic Module Rank

The improved rank dependence becomes particularly useful when the module rank grows logarithmically with the ambient dimension. In this regime, exact SVP and CVP admit running times with leading factors $(\log N)^{N/(2\mathrm{e})}$ and $(\log N)^{N/2}$, respectively.

This setting also arises through restriction of scalars. If an ideal lattice over a field K of degree N is viewed as a module over a subfield K' of degree N/r, its module rank becomes r. When K' is totally real or CM with bounded root discriminant, the algorithms of this section apply.

For $2 \le r = O(N/\log N)$, combining Theorem 6.2 with Kannan's exact SVP algorithm [Kan83, HS07] gives running time

$$\mathrm{poly}(B)\exp\left(\frac{N}{2\mathrm{e}}\left(\log r + \frac{\log(N/r)}{r}\right) + O(N) + o\left(\frac{N}{r}\log\frac{N}{r}\right)\right).$$

The leading expression is minimized at $r = W(\mathrm{e}N) \sim \log N$, where W denotes the Lambert W function. Thus, whenever a suitable subfield yields rank $r = \Theta(\log N)$, the SVP running time becomes $\mathrm{poly}(B)(\log N)^{N/(2\mathrm{e})+o(N)}$.

Corollary 6.8. *Let M be a module of rank $n = \Theta(\log N)$ over a totally real or CM field with $\rho = O(1)$, in the input model of Section 2.2, where $N = nd$.*

Exact SVP in M can be solved deterministically in polynomial space and time

$$\mathrm{poly}(B)(\log N)^{N/(2\mathrm{e})+o(N)}.$$

Exact CVP for every rational target, including enumeration of all closest vectors, can be solved deterministically in polynomial space and time

$$\mathrm{poly}(B, \ell(t))(\log N)^{N/2+o(N)}.$$

Proof. Since $d = N/n = \Theta(N/\log N)$, an exact SVP-oracle call in dimension at most $d+1$ takes time $d^{O(d)} = 2^{O(N)}$ using Kannan's algorithm. Moreover, $(N+6)^{O(n)} = 2^{O(\log^2 N)}$ and $\rho^{O(N)} = 2^{O(N)}$.

The SVP bound follows from (32), since $n^{N/(2\mathrm{e})} = (\log N)^{N/(2\mathrm{e})+o(N)}$. For CVP, (39) and $(n!)^{d/2} \le n^{N/2}$ give the claimed bound. In both cases, $2^{O(N)} = (\log N)^{o(N)}$, and the space bounds follow from the corresponding reductions. □

6.5 Growing Rank via Subfield Descent

The squared-radius allocation and block-minimum analysis do not depend on how the rank-one search routines are implemented. We can therefore replace the SVP-oracle-based enumeration of Section 4.1 by the subfield-descent routines of Section 3.2.

This combines the bounded-step subfield chains of Section 3 with the improved rank dependence established above. In particular, it extends the fixed-rank guarantees of Theorems 3.4 and 3.5 to growing module rank.

Theorem 6.9. *Fix s_0, d_0, and C_0. Let K be a totally real or CM field of degree d, with $\mathrm{rd}(K) \le C_0$, admitting a subfield chain as in Theorem 3.4. Let M be a rank-n module over $\mathcal{O}_K$, where $n \ge 2$ is part of the input and $N = nd$.*

Exact SVP in M can be solved deterministically in polynomial space and time

$$(N+6)^{n+2}\, 2^{O(N)} n^{N/(2\mathrm{e})}\, \mathrm{poly}(B, N),$$

where the implied constants depend only on s_0, d_0, C_0.

Proof. If the subfield chain is not supplied, compute it once using Lemma 3.6(i), in time $d^{O(\log d)}\,\mathrm{poly}(B, d)$, and reuse it throughout the algorithm.

Run Algorithm 1 with squared-radius allocation as in Theorem 6.2. Replace each rank-one SVP-oracle enumeration by the routine EN_0 of Section 3.2, and compute block minima using SV_0. In a halving search, the first block is queried in the original unscaled lattice, with the radius adjusted accordingly, as in the subfield-descent construction.

By Theorem 3.1, a block search at radius $r_j(w)$ costs at most $F_0 \max\{1, r_j(w)/H_j\}^d$, up to polynomial input-length factors, and outputs at most $(5\max\{1, r_j(w)/H_j\})^d$ points. As in Lemma 3.2, summing backward searches over the allocations gives total cost at most

$$r\, 4^r (5\sqrt{2})^{rd} F_0 \prod_{j=1}^{r} \max\left\{1, \frac{R}{\sqrt{r}\, H_j}\right\}^d.$$

Thus the analysis of Theorem 6.2 applies with the block-enumeration factor $16d$ replaced by F_0.

In particular, Lemma 6.5 bounds each rank-r search by $F_0 E_r$, where E_r is the search bound in the proof of Theorem 6.2. The preparation recurrence therefore yields time at most

$$(N+6)^{n+2}\, 2^{O(N)} n^{N/(2\mathrm{e})} \rho^{N/\sqrt{\mathrm{e}}} F_0\, \mathrm{poly}(B, N).$$

Since $\rho \le C_0$ and $F_0 = 2^{O(d)}$ by Theorem 3.1, the discriminant and descent factors are absorbed into $2^{O(N)}$.

All block data, centers, and squared radii have encoding length polynomial in B and N by Lemma 4.8. The arithmetic and storage requirements of the descent routines are bounded by Lemma 3.3. A depth-first traversal keeps only polynomially many suspended enumeration states, proving the space bound. □

For fixed n, this recovers the $2^{O(N)}\,\mathrm{poly}(B, N)$ bound of Theorem 3.4. More generally, if $n = (\log N)^{\gamma + o(1)}$ with $0 \le \gamma < 1$, then the running time becomes

$$(\log N)^{\gamma N/(2\mathrm{e}) + o(N)}\, \mathrm{poly}(B).$$

For these fields, restricting scalars to a suitable subfield and applying Corollary 6.8 gives the larger exponent $N/(2\mathrm{e})$ whenever $\gamma < 1$ and a corresponding logarithmic-rank restriction is available.

The same replacement of the rank-one enumeration routine gives an improved CVP bound.

Corollary 6.10. *Under the hypotheses of Theorem 6.9, exact CVP for every rational target t, including enumeration of all closest vectors, can be solved deterministically in polynomial space and time*

$$(N+6)^{O(n)}\, 2^{O(N)} (n!)^{d/2}\, \mathrm{poly}(B, N, \ell(t)),$$

where the implied constants depend only on s_0, d_0, C_0.

For $n = (\log N)^{\gamma + o(1)}$ with $0 \le \gamma < 1$, this becomes

$$(\log N)^{\gamma N/2 + o(N)}\, \mathrm{poly}(B, \ell(t)).$$

Proof. First compute a reduced flag using the algorithm of Theorem 6.9. Run Algorithm 3 on this flag, replacing each SVP-oracle ball enumeration by EN_0. Since the descent routine enumerates every point in the prescribed ball, correctness follows from Theorem 6.7.

Consider a rank-r prefix query with pivot j and $k = r - j + 1$ searched blocks. For an allocation w, let R_i be the block radius and $\overline{R}_i$ its auxiliary upper bound from Lemma 6.6. Write $a_i = \sqrt{d}\, G_i$. By Lemma 2.1,

$$\max\left\{1, \frac{R_i}{H_i}\right\} \le \sqrt{\rho}\,\frac{\overline{R}_i}{a_i}.$$

The determinant analysis of Lemma 6.6 gives

$$\prod_{i=j}^{r} \frac{\overline{R}_i}{a_i} \le (2\sqrt{2}\,\widehat{\rho}\sqrt{r})^k.$$

Using Theorem 3.1, one allocation costs at most $kF_0 \prod_{i=j}^{r}(5\max\{1, R_i/H_i\})^d$, up to the common polynomial factor. Since $\widehat{\rho} < (5/4)\rho$, this is at most

$$kF_0(18\rho^{3/2}\sqrt{r})^{kd}.$$

Summing over at most 4^k allocations bounds the projected-enumeration cost by

$$E_{r,k} = 4^k kF_0(18\rho^{3/2}\sqrt{r})^{kd}.$$

The number of accepted projected candidates remains bounded by $X_{r,k} = (1024\rho\sqrt{r})^{kd}$ from Lemma 6.6, independently of the enumeration routine.

Let T_r denote the cost of finding the minimum distance and streaming all closest vectors for a rank-r prefix. The two-pass recursion gives $T_r \le 2E_{r,k} + 3X_{r,k}T_{r-k}$. Set $Y_{r,k} = (1024\rho^{3/2}\sqrt{r})^{kd}$. Since $E_{r,k} \le NF_0Y_{r,k}$ and $X_{r,k} \le Y_{r,k}$, we obtain

$$\frac{T_r}{F_0} \le (2N+3)Y_{r,k}\max\left\{1, \frac{T_{r-k}}{F_0}\right\}.$$

Unfolding this recurrence along the successive prefix ranks, and using the factorial estimate from the proof of Theorem 6.7, gives

$$T_n \le F_0(2N+3)^n(1024\sqrt{\mathrm{e}})^N\rho^{3N/2}(n!)^{d/2}.$$

Since $\rho \le C_0$ and $F_0 = 2^{O(d)}$, this is bounded by $(N+6)^{O(n)}2^{O(N)}(n!)^{d/2}$ up to polynomial input-length factors. The preparation cost is absorbed by the same bound, using $n^{N/(2\mathrm{e})} \le n^{N/2} \le \mathrm{e}^{N/2}(n!)^{d/2}$.

The encoding-length and polynomial-space guarantees follow from Lemmas 4.8 and 3.3. A depth-first traversal stores only the active allocations and suspended descent states. Finally, for $n = (\log N)^{\gamma+o(1)}$ with $0 \le \gamma < 1$, the claimed asymptotic bound follows from $(N+6)^{O(n)} = 2^{o(N)}$ and $(n!)^{d/2} \le n^{N/2}$. □

Acknowledgments. We thank Martin Albrecht and Damien Stehlé for pointing out errors in an earlier version of this work and for sharing their detailed analysis.

A Bounds on Products of Block Minima

A.1 Bounds on $\mathcal{L}_m(\rho)$

Proof of Lemma 6.3. For $m = 1$ the assertion follows from the definition. For real $t \geq 2$, let $f(t) = t\log(t\rho)/(2(t-1))$. The numerator of $f'(t) = \big(t - 1 - \log(t\rho)\big)/\big(2(t-1)^2\big)$ is strictly increasing in t, so f' changes sign at most once, from negative to positive. Hence f has no interior maximum on $[2, m]$, and $\max_{2\leq k\leq m} f(k) = \max\{f(2), f(m)\}$. Since $f(2) = \log(2\rho)$, this maximum is $\mathcal{L}_m(\rho)$, and it is nondecreasing in m. For $m \geq 2$, the inequalities $\log m/(m-1) \leq \log 2$ and $m/(m-1) \leq 2$ give $f(m) \leq \log\rho + \frac{1}{2}\log(2m)$, and $f(2) \leq \log(\rho\sqrt{2m})$. □

A.2 Combining averages over intervals

The point count depends on products of block minima over index sets that need not be consecutive, whereas (7) controls only consecutive intervals. We adapt the arbitrary-subset analysis of Hanrot and Stehlé [HS07, Section 4.3]. Taking logarithms turns products into sums, and a piecewise constant extension of the logarithms to $[0, m]$ allows intervals with real endpoints to be enlarged and merged.

Lemma A.1. *Let x be constant on each cell $(i-1, i]$, $1 \leq i \leq m$, and let $A_b(s) = (b-s)^{-1}\int_s^b x(u)\,du$ for integer $b \leq m$. Suppose, for a constant $\mathcal{L} \geq 0$, that*

$$A_b(t) \geq A_b(s) + \mathcal{L}\log\frac{b-t}{b-s} \qquad (0 \leq s \leq t < b). \tag{40}$$

For a union I of whole cells, there are integer endpoints $0 = a_0 < a_1 < \cdots < a_q \leq m$, counts $k_j = |I \cap (a_{j-1}, a_j]|$ with $\sum_j k_j = |I|$, and nonincreasing positive densities $\delta_j = k_j/(a_j - a_{j-1}) \leq 1$, such that

$$\int_I x \geq \sum_{j=1}^{q} \delta_j \int_{a_{j-1}}^{a_j} x + \mathcal{L}\sum_{j=1}^{q} k_j \log\delta_j. \tag{41}$$

Proof. For an integer $c \in [0, m]$ we prove the same statement for unions $I \subseteq (c, m]$ of whole cells, with partition starting at $a_0 = c$. The lemma is the case $c = 0$. Since $\delta_j \int_{a_{j-1}}^{a_j} x = k_j A_{a_j}(a_{j-1})$, the right side of (41) equals $\sum_j k_j\big(A_{a_j}(a_{j-1}) + \mathcal{L}\log\delta_j\big)$. We induct on the number of maximal runs of selected cells. For $I = \emptyset$ take $q = 0$.

Let $(u, v]$ be the leftmost run of I, and apply the induction hypothesis to $I \cap (v, m]$ with starting point v. This gives consecutive intervals $(a_{j-1}, a_j]$, $j \geq 1$, with $a_0 = v$ and nonincreasing densities. We merge the run into them while maintaining a current interval $(s, b]$ with $c \leq s \leq u$ and $b = a_J$ for some $J \geq 0$, with $P = |I \cap (u, b]|$ selected cells and density $\delta = P/(b-s)$, such that

$$\int_{I\cap(u,m]} x \geq P\big(A_b(s) + \mathcal{L}\log\delta\big) + \sum_{j>J} k_j\big(A_{a_j}(a_{j-1}) + \mathcal{L}\log\delta_j\big) \tag{42}$$

and $\delta \geq \delta^+$, where $\delta^+ = \delta_{J+1}$, or $\delta^+ = 0$ if $J = q$. Initially $(s, b] = (u, v]$, $J = 0$, $P = v - u$ and $\delta = 1$. Since $PA_v(u) = \int_u^v x$, (42) is the induction hypothesis.

If $P/(b-c) \geq \delta^+$, apply (40) with the points $c \leq s$. Since $\delta(b-s)/(b-c) = P/(b-c)$, the first term on the right of (42) is at least $P\big(A_b(c) + \mathcal{L}\log(P/(b-c))\big)$. The cells of $(c, u]$ are not selected, so $(c, b]$ contains P selected cells and has density $P/(b-c) \in [\delta^+, 1]$. It becomes the first interval of the partition, and the process stops.

Otherwise $P/(b-c) < \delta^+ \leq \delta$. Let $s' = b - P/\delta^+$, so that $c < s' \leq s$. Applying (40) with the points $s' \leq s$, and using $(b-s)/(b-s') = \delta^+/\delta$, we bound the first term below by

$P\big(A_b(s') + \mathcal{L}\log\delta^+\big)$. The interval $(a_J, a_{J+1}]$ contains k_{J+1} selected cells and has the same density δ^+, so $PA_b(s') + k_{J+1}A_{a_{J+1}}(b) = \delta^+\int_{s'}^{a_{J+1}} x = (P + k_{J+1})A_{a_{J+1}}(s')$. Absorbing the summand for $j = J+1$ into the first term therefore gives (42) for $(s', a_{J+1}]$, with count $P + k_{J+1}$ and density δ^+, which is at least δ_{J+2}. Each such step removes one existing interval. When none remains, $\delta^+ = 0$ and the first case applies. The final endpoints are c and existing integer endpoints, and the counts are the numbers of selected cells in those intervals. This proves the claim. □

For example, let $m = 5$ and $I = (1,2] \cup (3,5]$. The run $(3,5]$ alone gives the interval $(2,5]$ of density $2/3$. The run $(1,2]$, extended to $c = 0$, would have density $1/2 < 2/3$, so the process extends it only to $s' = 2 - 3/2 = 1/2$, where its density equals $2/3$, and merges it with $(2,5]$. The merged interval $(1/2,5]$ contains 3 selected cells. Extending it to 0 gives density $3/5$ and $\int_I x \geq \frac{3}{5}\int_0^5 x + 3\mathcal{L}\log\frac{3}{5}$. The intermediate endpoint $1/2$ is not an integer. For this reason, (40) is stated for real $s \leq t$.

Prefix bounds control the first sum in (41), and convexity controls the second. Let $S(a) = \int_0^a x$ and $\delta_{q+1} = 0$. Summation by parts, with $S(a_0) = 0$, writes the first sum as $\sum_{j=1}^q (\delta_j - \delta_{j+1})S(a_j)$. Its coefficients are nonnegative. Hence, if $S(a_j) \geq \omega a_j - C$ at each endpoint for some real ω and $C \geq 0$, then the first sum is at least $\omega k - \delta_1 C$, where $k = |I|$, because $\sum_j (\delta_j - \delta_{j+1})a_j = k$ and $\sum_j (\delta_j - \delta_{j+1}) = \delta_1$. With $\ell_j = a_j - a_{j-1}$ and $a_q \leq m$, the convexity of $u\log u$ gives

$$\sum_j k_j \log\delta_j = \sum_j \ell_j\delta_j \log\delta_j \geq k\log(k/a_q) \geq k\log(k/m).$$

A.3 Products over arbitrary subsets

Proof of Lemma 6.4. We first derive (40) from (7). Let $x_i = \log h_i$, let $x = x_i$ on each cell $(i-1, i]$, and use the averages $A_b(s)$ of Lemma A.1. For $s \in (i-1, i)$, let $q = b - i + 1$, the number of cells in $(i-1, b]$. If $q \geq 2$, (7) on $\{i, \dots, b\}$ gives $x_i - A_b(i-1) \leq \frac{1}{2}\log(q\rho)$, while $b - s > b - i = q - 1$. Hence, if $x_i - A_b(i-1) < 0$ then $x_i - A_b(s) < 0$, and otherwise

$$x_i - A_b(s) = \frac{q}{b-s}\big(x_i - A_b(i-1)\big) \leq \frac{q\log(q\rho)}{2(q-1)} \leq \mathcal{L}_m(\rho),$$

where the last step is Lemma 6.3. For $q = 1$ the difference is zero. In all cases $A_b'(s) = (A_b(s) - x_i)/(b-s) \geq -\mathcal{L}_m(\rho)/(b-s)$, and integration proves (40) with $\mathcal{L} = \mathcal{L}_m(\rho)$.

Applying Lemma A.1 to the union of the cells $(i-1, i]$, $i \in I$, gives (41) with $\mathcal{L} = \mathcal{L}_m(\rho)$ and $\int_I x = \sum_{i\in I} x_i$. By (7) on $\{1, \dots, a\}$ and $a \leq m$, every prefix integral satisfies $\int_0^a x \geq a(x_1 - \frac{1}{2}\log(m\rho))$. The summation step of Appendix A.2 bounds the first sum below by $k(x_1 - \frac{1}{2}\log(m\rho))$, and the second sum below by $k\log(k/m)$. This proves the first inequality in (33), and the second follows from Lemma 6.3.

For (34), let I be the set of indices i with $h_1/(\sqrt{m}\,h_i) > 1$. We divide (33) by $m^{k/2}$ and use $k\log(m/k) \leq m/\mathrm{e}$ and $k/2 + k\log(m/k) \leq m/\sqrt{\mathrm{e}}$. The empty subset has product one. □

A.4 Allowing an unreduced first block

Proof of Lemma 6.5. The first block need not be reduced, so (7) is available only for the projected suffix. For prefixes that contain the first block, the hypothesis that every prefix radius is at least $R/2$ replaces it. Let $x_j = \log(H_j/R)$, so that $0 \leq x_1 \leq \log 2$. Since every prefix radius is at least $R/2$ and Lemma 2.1 gives $\sqrt{d}\,G_j \leq \sqrt{\rho}\,H_j$, we have $R/2 \leq \sqrt{a\rho}\,(\prod_{j\leq a} H_j)^{1/a}$, that is, $\sum_{j=1}^a x_j \geq -\frac{a}{2}\log(4a\rho)$

for $1 \le a \le r$. Applying this to the prefix of length $a+1 \le r$ and subtracting x_1 gives

$$\sum_{j=2}^{a+1} x_j \ge -\tfrac{a}{2}\log(4r\rho) - C, \qquad C = x_1 + \tfrac{1}{2}\log(4r\rho) \le \tfrac{1}{2}\log(r\rho) + 2\log 2.$$

Because the projected suffix is reduced, the positive numbers $H_2/R, \dots, H_r/R$ satisfy (7), and the first step of the proof of Lemma 6.4 yields (40) for their logarithms $x_2, \dots, x_r$, with $\mathcal{L} = \mathcal{L}_m(\rho)$. We apply Lemma A.1 to the cells of I. The summation step of Appendix A.2, now with the constant C, bounds the first sum below by $-\frac{k}{2}\log(4r\rho) - \delta_1 C$, which is at least $-\frac{k}{2}\log(4r\rho) - C$ since $\delta_1 \le 1$. Bounding the second sum as before gives $\sum_{j\in I} x_j \ge -\frac{k}{2}\log(4r\rho) - C - \mathcal{L}_m(\rho)\, k \log(m/k)$. Since $\log(R/(\sqrt{r}\, H_j)) = -x_j - \frac{1}{2}\log r$, this proves (35).

For (36), the first factor is one because $R \le H_1$. Let I be the set of $j \ge 2$ with $R/(\sqrt{r}\, H_j) > 1$, and let $g(k) = \frac{k}{2}\log\rho + \mathcal{L}_m(\rho) k \log(m/k)$. By (35), the product is at most $\mathrm{e}^C 2^k \mathrm{e}^{g(k)}$, and $\mathrm{e}^C \le 4\sqrt{r\rho}$. On the real interval $0 \le k \le m$, g is strictly concave since $g''(k) = -\mathcal{L}_m(\rho)/k < 0$. Its critical point $k = m\exp(\log\rho/(2\mathcal{L}_m(\rho)) - 1)$ lies below m, and its maximum is $\Psi_m(\rho)$. This proves the first bound. For the second, $k\log(m/k) = 0$ when $m = 1$, and $\frac{1}{2} \le 1/\sqrt{\mathrm{e}}$. For $m \ge 2$, we use $\mathcal{L}_m(\rho) \le \log(\rho\sqrt{2m})$, $k\log(m/k) \le m/\mathrm{e}$ and $k/2 + k\log(m/k) \le m/\sqrt{\mathrm{e}}$. □

B Arithmetic and Bit Lengths

B.1 The metric on a rank-one module

We derive (4) from compatibility of the inner product. For nonzero $v \in M$, the $\mathbb{Q}$-bilinear form $b(x,y) = \langle xv, yv\rangle$ on K satisfies $b(x,y) = \langle v, x^\star yv\rangle = b(1, x^\star y)$ by compatibility. The trace form $(x,y) \mapsto \mathrm{Tr}_{K/\mathbb{Q}}(xy)$ is nondegenerate, so $b(1,z) = \mathrm{Tr}_{K/\mathbb{Q}}(\beta z)$ for a unique $\beta \in K$, and since $\sigma(x^\star) = \overline{\sigma(x)}$, $b(x,x) = \mathrm{Tr}_{K/\mathbb{Q}}(\beta x^\star x) = \sum_\sigma \sigma(\beta)|\sigma(x)|^2$. Symmetry of b gives $\beta = \beta^\star$, so every conjugate of β is real. A negative conjugate would contradict positive definiteness of b, by continuity and the density of K in $K \otimes_{\mathbb{Q}} \mathbb{R}$. Since $\beta \ne 0$, no conjugate vanishes, so every $\sigma(\beta)$ is positive. This gives (4) with $w_\sigma = \sigma(\beta)$.

B.2 Intersections, projections, and lifts

Let V be a K-subspace of the rational span of M. Let U be the unimodular change-of-basis matrix from the input basis of M, with Gram matrix G, to an adapted basis whose first vectors span $M \cap V$, and partition

$$U^T G U = \begin{pmatrix} F & E \\ E^T & F' \end{pmatrix}, \qquad Q = F' - E^T F^{-1} E.$$

The block F is the Gram matrix of $M \cap V$, and Q is that of $\pi_{V^\perp} M$. For integer coefficient vectors x and y of the two groups, let $U(x,y)$ be the corresponding lattice vector, and let $(\tau_x, \tau_y) = U^{-1}t$ be the target coordinates, partitioned in the same way. We have

$$\|U(x,y) - t\|^2 = (x - \tau_x + F^{-1}E(y-\tau_y))^T F (x - \tau_x + F^{-1}E(y-\tau_y)) + (y-\tau_y)^T Q (y-\tau_y). \quad (43)$$

For a fixed choice of the second group of coordinates y, the second term is fixed. The first is a closest-vector query in $M \cap V$ with shifted target $\tau_x - F^{-1}E(y-\tau_y)$. The projection and lift maps use this identity to compute the target of each recursive branch and the squared distance contributed

by the second group. The matrix Q is the Schur complement, obtained by minimizing over the real coordinates of the first group before imposing their integrality.

The intersection is saturated: if $kx \in M \cap V$ for $x \in M$ and a nonzero integer k, then $x \in M \cap V$. The integer kernel of the linear conditions defining V therefore gives an exact basis of $M \cap V$, and a unimodular completion [KB79] extends it to a basis of M. The added vectors project to a basis of $\pi_{V^\perp} M$, so an integer combination of these projections lifts to the same combination of the added vectors. For nonzero $v \in M$, the lattice on the K-line Kv is the full intersection $M \cap Kv = Iv$, where the coefficient ideal $I = \{x \in K : xv \in M\}$ contains $\mathcal{O}_K$ but need not be principal. For example, the fractional ideal $(1, (1+\sqrt{-5})/2)$ of $\mathbb{Z}[\sqrt{-5}]$ contains $\mathbb{Z}[\sqrt{-5}]$ and is not principal.

For the dual split (6), take a basis of M^* adapted to Δ, so that its first d vectors span Δ, and let $b_1, \ldots, b_{rd}$ be the dual basis of M, obtained by inverting and transposing. The vectors $b_{d+1}, \ldots, b_{rd}$ are orthogonal to W and form a basis of L, the vectors $b_1, \ldots, b_d$ lift a basis of Λ, and the Gram matrix of $\Lambda = \Delta^*$ is the inverse of the Gram matrix of Δ.

B.3 Choice of the Rescaling Factor

We prove Lemma 2.4 and describe how to compute the rescaling factor using exact rational arithmetic.

Let V be the first K-line, and consider the Gram blocks F, E, Q of (43). Scaling lengths in the first block by a factor $0 < a \leq 1$ changes the Gram matrix to

$$G_a = \begin{pmatrix} a^2 F & a^2 E \\ a^2 E^T & Q + a^2 E^T F^{-1} E \end{pmatrix}. \tag{44}$$

The Schur complement remains Q, and the shifted target in the first block remains $\tau_x - F^{-1}E(y - \tau_y)$. Thus the projected suffix and its prepared flag can be reused.

Moreover, scaling the first block preserves its module structure and compatibility of the metric, so Lemma 2.1 continues to apply.

Proof of Lemma 2.4. Let $H = H_1$ be the minimum of the first block. Since this block is contained in M, we have $\lambda_1(M) \leq H$.

Suppose first that $R_j > H/2$ for every $j \geq 2$. Set $R = H$. The first prefix satisfies $R_1 \geq H$ by Minkowski's theorem, while the remaining prefix bounds exceed $R/2$. Moreover, $R = H_1$, and the first block contains a nonzero vector of length R. Hence all search conditions hold, and $M \cap B(0, H)$ contains a shortest vector of M.

Now suppose that $R_j \leq H/2$ for some $j \geq 2$. Scaling the first block by a changes the j-th prefix bound to

$$R_j(a) = a^{1/j} R_j.$$

Consequently,

$$\frac{R_j(a)}{aH} = a^{1/j - 1} \frac{R_j}{H}.$$

For $j \geq 2$, this ratio strictly increases as a decreases and tends to infinity as $a \to 0$.

Choose the largest integer $q \geq 0$ such that

$$\min_{j \geq 2} R_j(2^{-q}) \leq 2^{-q} H/2,$$

and put $a = 2^{-q}$. Such a q exists because the condition holds at $q = 0$ but fails for all sufficiently large q.

By maximality, every $j \geq 2$ satisfies

$$R_j(a/2) > aH/4.$$

Since $R_j(a) = 2^{1/j} R_j(a/2)$, we obtain

$$R_j(a) > aH/4 \qquad \text{for every } j \geq 2.$$

Together with the defining inequality, this proves (10).

Set $R = aH/2$ and choose a prefix attaining the minimum of the $R_j(a)$ for $j \geq 2$. Minkowski's theorem guarantees that this prefix contains a nonzero vector of scaled length at most R.

The scaled first block has minimum $aH = 2R$, and its Minkowski bound satisfies $R_1(a) \geq aH$. All other prefix bounds exceed $R/2$. Since the projected suffix remains reduced, the chosen prefix satisfies the search conditions at radius R.

A nonzero vector found by this search has scaled length at most $aH/2$. Undoing the scaling increases its length by at most $1/a$, so its original length is at most $H/2$. It cannot lie in the first K-line, whose scaled minimum is aH.

For each $i \geq 2$ in the selected prefix, the i-th projected block is the same as in the original flag. Its minimum is therefore H_i, and the reducedness properties of the projected suffix are preserved.

Exact computation of the scale. Let g_j be the determinant of the Gram matrix of the unscaled j-th prefix. Since that prefix has dimension jd and its determinant is multiplied by a^{2d}, we have

$$R_j(a)^{2jd} = (jd)^{jd} a^{2d} g_j.$$

Thus

$$R_j(a) \leq aH/2 \quad \Longleftrightarrow \quad (jd)^{jd} a^{2d} g_j \leq (a^2 H^2/4)^{jd}. \tag{45}$$

All quantities in this comparison are rational, so it can be evaluated exactly without computing roots.

It remains to bound the dyadic exponent q. For every prefix,

$$R_j \geq \lambda_1(M_j) \geq \lambda_1(M).$$

If $R_j(a) \leq aH/2$ for some $j \geq 2$, then

$$a^{1/j} \lambda_1(M) \leq aH/2,$$

which implies

$$a \geq \left(\frac{2\lambda_1(M)}{H} \right)^{j/(j-1)}.$$

In the present case, $2\lambda_1(M) \leq H$. Since $1 < j/(j-1) \leq 2$, we conclude that

$$a \geq \left(\frac{2\lambda_1(M)}{H} \right)^{2}.$$

At the beginning of the preparation of M, the LLL bound gives

$$H \leq 2^{(N-1)/2} \lambda_1(M), \qquad N = \dim M.$$

Subsequent halving passes only decrease H. Therefore

$$q = \log_2(1/a) = O(N).$$

The predicate in (45) is monotone in q. We find the largest admissible q by doubling a trial exponent until the predicate fails, followed by binary search. Since $q = O(N)$, the search requires polynomially many exact rational operations, and the factor $a = 2^{-q}$ has $O(N)$ bits. □

B.4 Bit Lengths

Proof of Lemmas 3.3 and 4.8. We establish uniform polynomial bounds on the encoding lengths of all intermediate lattices, bases, targets, and squared radii. The bounds depend polynomially on the original input length B and dimension N, and, for an enumeration query, on the encoding length of its target and squared radius. Their polynomial degrees are independent of the module rank.

All computations use exact rational arithmetic. We distinguish the construction of module instances from the temporary lattices used by the SVP-oracle reduction.

Representing recursive module instances. Let $M_+ = M$ and $M_- = M^*$, each with a fixed basis in the original rational coordinates. For rational subspaces $X \subseteq Y$ that are stable under the relevant field action, write

$$L^{\varepsilon}_{X,Y} = \pi_{X^\perp}(M_\varepsilon \cap Y), \qquad \varepsilon \in \{+,-\}.$$

The lattice is viewed in the Euclidean space $Y \cap X^\perp$.

Every unscaled module instance arising in the recursive algorithms has this form. Splitting off a module line replaces one of the subspaces by its extension by that line, while restricting scalars changes only the field over which the same lattice is regarded as a module.

The family is also closed under duality. The standard intersection–projection duality gives

$$(L^{\varepsilon}_{X,Y})^* = L^{-\varepsilon}_{Y^\perp,X^\perp}.$$

Thus dualization can be represented by switching between the two fixed input lattices and taking orthogonal complements, rather than by repeatedly dualizing intermediate basis matrices.

We maintain rational flags in the fixed input spaces, together with adapted integral bases. Each flag includes the zero subspace, the full ambient space, and the subspaces defining the current module instance. When a module line is selected, the flag is refined; under dualization, the corresponding flag is replaced by the reversed flag of orthogonal complements.

The temporary scaled metrics used in halving searches are not stored as recursive module instances. As explained in Appendix B.3, their block searches are implemented in the original compatible metrics, with adjusted radii.

Bounds on successive minima. We first bound the first and last successive minima of all blocks appearing in these flags, together with their reciprocals.

Let $T_0 \geq 1$ bound these quantities for the original lattices M and M^*. Such a bound can be chosen with $\log T_0 \leq \mathrm{poly}(B,N)$, since their Gram matrices are rational, have polynomial encoding length, and are positive definite.

Let $\rho_{\max} \geq 1$ bound the root discriminants of all fields used in the recursion. Let $\xi \geq 1$ bound the operator norms of multiplication by the elements of the integral bases used at these levels, including their adjoints. The field arithmetic and subfield representations have polynomial encoding length, and hence

$$\log \rho_{\max} + \log(1+\xi) \leq \mathrm{poly}(B,N).$$

The same operator-norm bound applies to invariant subspaces and their orthogonal projections, since the field actions commute with these projections.

Consider a module instance J of rank at least two and a selected first K-line. Let H be the minimum of the rank-one block on this line. The line is initialized using LLL and subsequently replaced only by shorter lines during halving searches. Consequently,

$$\lambda_1(J) \leq H \leq 2^{N/2}\lambda_1(J).$$

Let J_1 be the rank-one block on this line and J_2 the projected suffix. We have $\lambda_1(J_1) = H$. Multiplying a shortest vector of J_1 by an integral basis of the field gives linearly independent vectors in J_1. Therefore

$$\lambda_{\dim J_1}(J_1) \le \xi H \le \xi 2^{N/2} \lambda_{\dim J}(J).$$

Orthogonal projection cannot increase vector lengths, so

$$\lambda_{\dim J_2}(J_2) \le \lambda_{\dim J}(J).$$

To bound the minimum of J_2, let $H_2 = \lambda_1(J_2)$. Consider the rank-two module obtained from the first line and the preimage of the line generated by a shortest vector of J_2. Its first minimum is at least $\lambda_1(J)$. Applying Minkowski's theorem and Lemma 2.1 to its two blocks gives

$$\lambda_1(J)^2 \le 2\rho_{\max} H H_2.$$

Hence

$$H_2 \ge \frac{\lambda_1(J)^2}{2\rho_{\max} H} \ge \frac{\lambda_1(J)}{2^{N/2+1}\rho_{\max}}.$$

Thus splitting off a module line changes the first and last successive minima of the resulting blocks, and their reciprocals, by at most a factor

$$c_{\text{split}} = 2^{O(N)}(1+\rho_{\max})(1+\xi).$$

Dualization is controlled by transference. For any lattice J of dimension at most N, the standard transference bounds give

$$\lambda_1(J^*) \ge \frac{1}{\lambda_{\dim J}(J)}, \qquad \lambda_{\dim J}(J^*) \le \frac{2N}{\lambda_1(J)}.$$

Thus dualization changes the corresponding bounds by a factor of at most $2N$.

Along any active recursive construction branch, at most $O(N)$ module lines are selected or removed, and at most $O(N)$ dualizations occur. A halving search restarts the preparation of the same module instance and does not accumulate additional flag refinements. Restricting scalars does not change the underlying lattice or its successive minima.

It follows that the first and last successive minima of every block, and their reciprocals, lie in an interval

$$[T^{-1}, T], \qquad T = T_0 c^{O(N)}.$$

Here $c = 2^{O(N)}(1+\rho_{\max})(1+\xi)$. In particular, $\log T \le \text{poly}(B, N)$.

Polynomial-size adapted bases. We now use these geometric bounds to control the integral representations of the recursive lattices.

Consider a maintained flag

$$0 = V_0 \subsetneq V_1 \subsetneq \cdots \subsetneq V_q = \text{span}_{\mathbb{R}}(M_\varepsilon).$$

Its blocks are

$$J_i = \pi_{V_{i-1}^{\perp}}(M_\varepsilon \cap V_i).$$

Every intersection $M_\varepsilon \cap V_i$ is saturated in M_ε. Consequently, bases of the successive quotients can be lifted to an adapted integral basis of M_ε.

By the preceding bounds, the last successive minimum of every J_i is at most T. LLL therefore gives a basis of each block whose vectors have lengths at most $V = 2^{O(N)}T$.

Choose integral lifts of the block bases, proceeding through the flag. For a basis vector of J_i, reduce its lift against the previously chosen integral lifts of the bases of $J_{i-1}, \dots, J_1$, in that order. At block J_j, subtract an integral combination of its lifted basis vectors so that the coefficients of the corresponding orthogonal component lie in $[-1/2, 1/2)$. This changes only components in $J_1, \dots, J_j$ and preserves the image in J_i.

After these reductions, each earlier block component lies in the centered half-open fundamental parallelepiped of its block basis. Its norm is therefore at most one half of the sum of the lengths of the corresponding basis vectors. Since the block components are orthogonal and their total dimension is at most N, every vector of the resulting adapted basis has norm at most

$$O(N^2)V = 2^{O(N)}T.$$

The Gram matrices of M and M^*, in their fixed input bases, have smallest eigenvalues at least $2^{-\operatorname{poly}(B,N)}$. Therefore the integral coordinates of these adapted basis vectors have absolute values at most $2^{\operatorname{poly}(B,N)}T$. In particular, the adapted basis constructed in this way has polynomial encoding length.

The subspaces in the flag are spanned by initial segments of this basis. They consequently admit rational coordinate descriptions of polynomial encoding length.

Conversely, given these rational descriptions, the required intersections and adapted bases can be reconstructed from the fixed input bases using integer kernels, Hermite normal form, and unimodular completion [KB79]. LLL reduction of the quotient blocks and size reduction of their lifts recover adapted bases with the bounds above.

All rational matrices involved have polynomial encoding length: integer kernel and normal-form computations have polynomial bit complexity, and Hadamard bounds control the minors and denominators arising in the corresponding rational linear algebra.

After each line selection or dualization, we reconstruct the adapted basis from the fixed input representation and discard the previous one. Thus the algorithm does not accumulate products of successive change-of-basis matrices.

Targets, radii, and oracle instances. The Gram matrices of all stored blocks are obtained by rational basis changes and Schur complements of polynomial-bit matrices. Their entries, determinants, and inverses therefore have polynomial encoding length.

Fix $C = \operatorname{poly}(B, N)$ large enough to bound the encoding lengths of these Gram matrices and the matrices used for coordinate changes and target updates, and to ensure that every block Gram matrix has smallest eigenvalue at least 2^{-C}. Write t and $s \geq 0$ for the current target and squared radius. If an integer coefficient vector z is enumerated in this ball, then

$$\|z\|_\infty \leq \|t\|_\infty + 2^C\sqrt{s}.$$

The coordinate changes and shifted-target identity (43) express each target update as an affine map in the current target and the integral branch coordinates, with coefficients of uniformly polynomial encoding length. Residual squared radii are obtained by subtracting nonnegative squared distances. The rescalings described below and the allocation factors in Section 6 have polynomial encoding length. Radii introduced by the SVP and CVP routines have polynomial encoding length by the preceding bounds on lattice minima and shortest-vector coordinates. Consequently, the maximum of 1, $\|t\|_\infty$, and $\sqrt{s}$ increases by at most a factor $2^{\operatorname{poly}(B,N)}$ per level; the same bound controls the enumerated integer coordinates. There are only $O(N)$ levels along an active branch.

To control denominators, represent each target using a common denominator. An affine target update multiplies this denominator only by denominators of the transformation matrices; the integral branch coordinates introduce no additional denominator. Thus its bit length increases by at

most poly(B, N) per level. Each squared-radius update is linear in a previously available squared radius and subtracts a quadratic form in a target and integral branch coordinates. Expanding these updates since the most recent initialization expresses each squared radius as a rationally weighted sum of the initial squared radius and $O(N)$ squared-distance contributions. The weights have polynomial encoding length, and each contribution uses a target denominator only to the second power. Consequently, the denominators of the squared radii also have polynomial bit length.

Combining the magnitude and denominator bounds shows that all targets, squared radii, enumerated coefficient vectors, and their size-reduced integral lifts have encoding lengths polynomial in B, N, and the original query length.

For a halving search, the rescaling factor is $a = 2^{-q}$ with $q = O(N)$ by Appendix B.3. The corresponding block queries use the original metric with the squared radius divided by a^2. This increases its encoding length by only $O(N)$ bits.

For CVP, the search radius is computed from a shortest dual vector. The vector has polynomial-bit coordinates by the preceding bounds, so its squared norm and the resulting squared search radius also have polynomial encoding length.

Finally, consider the temporary lattices used by the SVP-oracle implementation. For an m-dimensional rational Gram matrix with polynomial encoding length, a positive lower eigenvalue bound gives a polynomial-bit upper bound on the number of lattice points in any ball whose squared radius has polynomial encoding length. Thus the primes used to construct the index-p sublattices in Lemma 4.2 have polynomial bit length.

The corresponding sublattice bases and augmented Gram matrices are constructed by rational arithmetic on polynomial-bit data. They therefore satisfy the same uniform encoding-length bounds.

Polynomial working space. All enumeration procedures are implemented depth-first. Only the active recursive branch, the states of suspended searches, and one output candidate are stored.

At a subfield step of relative degree ν_j, at most ν_j lower-level states are simultaneously active or suspended. If S_j is the working space at field level j and P bounds the local polynomial storage, then

$$S_j \le \nu_j S_{j+1} + P.$$

Since $\prod_j \nu_j \le N$, this recurrence gives polynomial working space.

The module-rank recursion also has depth at most N and stores only polynomially many additional intermediate objects.

All stored objects have polynomial encoding length, uniformly over the module rank and field levels. This proves the encoding-length and polynomial-space claims of Lemmas 3.3 and 4.8. □

C The Local–Global Reduction of Tian, Sun and Zhu

The reduction of [TSZ14] takes a totally real field $E = F(\alpha)$ and a fractional ideal A of E, where $F \subset \mathbb{R}$ is fixed and α has a monic minimal polynomial $f \in \mathcal{O}_F[t]$. It minimizes $\mathrm{Tr}_{E/F}(y^2)$ over nonzero $y \in A$ [TSZ14, Section 4]. It selects a prime P of F that is unramified in E, not inert, and coprime to the index $[\mathcal{O}_E : \mathcal{O}_F[\alpha]]$. With F_P the completion of F at P, it decomposes $E \otimes_F F_P$ into local fields $E_1, \dots, E_r$ of degrees $d_i = [E_i : F_P]$, chooses number fields E_i' with $[E_i' : F] = d_i$, and recurses on ideals $A_i \subset E_i'$. Its main theorem [TSZ14, Theorem 2] bounds the running time polynomially in the costs of four oracles [TSZ14, Section 4.3]: generating a random prime of $\mathcal{O}_F$, testing primality of an ideal of $\mathcal{O}_E$, and computing the relative discriminant and the index. For fixed F and a given integral basis of $\mathcal{O}_E$, each oracle runs in polynomial time, and only the first is randomized. By the Dedekind–Kummer theorem, the primality test reduces to an irreducibility test for f modulo P [Rab80]. Since Lemma E.1 recovers the integral basis of an ideal lattice in

polynomial time, [TSZ14, Theorem 2] as stated would give a randomized polynomial-time algorithm for exact SVP on every fractional ideal of a totally real field in the canonical norm.

The correctness of the reduction rests on the chain of equalities on p. 389 of [TSZ14], which equates $\min\{\mathrm{Tr}_{E/F}(y^2) : 0 \neq y \in A\}$, via a minimum of sums of local traces in F_P, with $\min_i \mathrm{Tr}_{E_i'/F}(x_i^{*2})$ for shortest vectors x_i^* of the ideals $A_i \subset E_i'$. The chain already fails in degree two. Take $F = \mathbb{Q}$, $E = \mathbb{Q}(\sqrt{2})$, $\alpha = \sqrt{2}$ and $A = \mathbb{Z}[\sqrt{2}] = \mathcal{O}_E$. A prime of $\mathbb{Q}$ that is unramified and not inert in E splits, so $r = 2$, $d_1 = d_2 = 1$ and $E_1' = E_2' = \mathbb{Q}$. For the minimizing index m, the last expression of the chain is the square x_m^{*2} of a rational number $x_m^* \in A_m \subset \mathbb{Q}$. But $\mathrm{Tr}_{E/\mathbb{Q}}((a+b\sqrt{2})^2) = 2a^2 + 4b^2$, so the first expression equals 2, attained at $y = 1$. Since 2 is not a rational square, it cannot equal the last expression.

D Explicit Root-Discriminant Dependence

Theorem D.1. *Assume the hypotheses of Theorem 3.4, except that $C_0 \geq \max\{1, \mathrm{rd}(K_0)\}$ may be any real number. Let M have rank $n \leq n_0$ over $\mathcal{O}_{K_0}$ and dimension N. Exact SVP can be solved in time*

$$C_0^{\frac{n+1}{2n}N} 2^{O(N)} \,\mathrm{poly}(B, N),$$

and exact CVP, including enumeration of all closest vectors, in time

$$C_0^{(1+\frac{1}{n})N} 2^{O(N)} \,\mathrm{poly}(B, N, \ell(t)).$$

The implied constants and polynomial factors depend only on n_0, s_0, d_0.

Proof of Theorem D.1. The algorithms do not use C_0, which appears only in the complexity analysis. If $h = 0$, the dimension is at most $n_0 d_0$, and direct enumeration gives the claimed bounds. Hence assume $h \geq 1$.

We first track the dependence on C_0 in the rank-one routines of Theorem 3.1. At level $j < h$, write $s = \nu_j$ and $d = D_{j+1}$, so that $D_j = sd$. The constant in (9) satisfies

$$C_s^d = 2^{D_j + d} s^{D_j/2} C_0^{D_j/2}.$$

Since $s \leq s_0$ is fixed, the preparation and enumeration bounds in the proof of Theorem 3.1 yield

$$F_j \leq C_0^{D_j/2} 2^{O(D_j)} F_{j+1},$$

where the implied constant is independent of C_0, and uniform polynomial factors in the input length are suppressed.

Using $F_h = O(1)$ and $\sum_{k \geq j} D_k \leq 2D_j$, we obtain

$$F_j \leq C_0^{\frac{1}{2}\sum_{k=j}^{h-1} D_k} 2^{O(D_j)} \leq C_0^{D_j} 2^{O(D_j)}.$$

SVP. Let M have rank n over K_0 and dimension $N = nD_0$. Over K_1, it has rank $r = n[K_0 : K_1] \leq n_0 s_0$ and dimension $N = rD_1$. Applying the preparation analysis at this level contributes a factor $C_0^{N/2}$, apart from $2^{O(N)}$ and uniform polynomial factors. Together with the bound $F_1 \leq C_0^{D_1} 2^{O(D_1)}$, this gives

$$C_0^{N/2 + D_1} 2^{O(N)} \,\mathrm{poly}(B, N).$$

Since $[K_0 : K_1] \geq 2$, we have $D_1 \leq D_0/2 = N/(2n)$. Therefore

$$\frac{N}{2} + D_1 \leq \frac{n+1}{2n} N,$$

which proves the SVP bound.

CVP. We use the rank-recursion analysis in the proof of Theorem 3.5. At rank $r \leq n$, the number of projected candidates is bounded by $Z = (9nC_0)^{D_0}$. The corresponding enumeration cost satisfies

$$E_r \leq F_0 (2nC_0)^{D_0} \operatorname{poly}(B, N, \ell(t)) \leq C_0^{2D_0} 2^{O(N)} \operatorname{poly}(B, N, \ell(t)).$$

By the SVP bound just proved, the dual-SVP cost at rank r satisfies

$$S_r \leq C_0^{(r+1)D_0/2} 2^{O(N)} \operatorname{poly}(B, N, \ell(t)).$$

The recurrence $Q_r \leq 2(S_r + E_r) + 3ZQ_{r-1}$ can be unfolded over the at most n rank levels. The contribution of an enumeration at rank r has C_0-exponent at most $(n - r + 2)D_0$, while that of a dual-SVP call has exponent at most

$$\left(n - r + \frac{r+1}{2}\right) D_0.$$

Both are at most $(n+1)D_0$. The rank-zero base case satisfies the same bound. Since $D_0 = N/n$, we conclude that

$$Q_n \leq C_0^{(1+1/n)N} 2^{O(N)} \operatorname{poly}(B, N, \ell(t)).$$

This proves the CVP bound. □

E Recovering Maximal-Order Arithmetic for Ideal Lattices

The results of Section 3.4 assume that the integral arithmetic of the underlying number field is given. We show that this assumption can be removed for ideal lattices specified by a defining polynomial and a rational ideal basis.

We first recover the maximal-order arithmetic from the action of a field generator.

Lemma E.1. *Let $A \in \mathbb{Q}^{N \times N}$ have an irreducible minimal polynomial of degree d, and let $K = \mathbb{Q}[A]$. Suppose $\mathbb{Z}^N$ is stable under the action of $\mathcal{O}_K$. Given A and a positive definite rational Gram matrix G compatible with the field action as in Section 2.2, one can recover an integral basis of $\mathcal{O}_K$, its multiplication and action matrices, its discriminant, and its involution in polynomial time, with polynomial encoding length.*

Proof. Every element of K is a rational linear combination of $I, A, \ldots, A^{d-1}$. Let C be the full-column-rank matrix whose columns are

$$\operatorname{vec}(I), \operatorname{vec}(A), \ldots, \operatorname{vec}(A^{d-1}).$$

The coefficient vectors of field elements preserving $\mathbb{Z}^N$ form the lattice

$$\{x \in \mathbb{Q}^d : Cx \in \mathbb{Z}^{N^2}\},$$

which is the dual of the lattice generated by the rows of C.

After clearing denominators, compute a basis of this row lattice by Hermite normal form [KB79]. Its rational dual gives a basis of

$$R_M = \{\alpha \in K : \alpha\mathbb{Z}^N \subseteq \mathbb{Z}^N\},$$

together with the corresponding integral action matrices.

Every element of R_M has an integral action matrix and therefore satisfies a monic polynomial over $\mathbb{Z}$. Hence $R_M \subseteq \mathcal{O}_K$. Conversely, the assumed stability under $\mathcal{O}_K$ implies $\mathcal{O}_K \subseteq R_M$. Thus $R_M = \mathcal{O}_K$.

The representation is faithful, and $\mathbb{Q}^N$ has dimension $n = N/d$ over K. Consequently, the field trace of an element is $1/n$ times the trace of its action matrix. The determinant of the resulting trace pairing gives the discriminant. Multiplication and the corresponding action matrices are recovered by exact linear algebra in the power basis.

Finally, compatibility with G implies

$$G^{-1}A^TG \in \mathbb{Q}[A].$$

Expressing this adjoint in the power basis recovers the involution $\star$. All computations have polynomial bit complexity. □

Combining this recovery procedure with the subfield-chain search of Section 3.4 extends our algorithms to ideal lattices without a supplied maximal-order basis.

Corollary E.2. *Fix integers $s_0 \geq 2$, $d_0 \geq 1$, and a rational constant $C_0 \geq 1$. Let $K = \mathbb{Q}(\theta)$ be a totally real or CM field given by an irreducible defining polynomial, and let I be a nonzero fractional $\mathcal{O}_K$-ideal given by a rational basis in the power basis of K.*

Suppose a compatible positive definite rational Gram matrix is supplied, $\mathrm{rd}(K) \leq C_0$, and K admits a subfield chain with step degrees at most s_0 and bottom degree at most d_0.

Then exact SVP on I and exact CVP for every rational target, including enumeration of all closest vectors, can be solved deterministically in time

$$2^{O(D)}\,\mathrm{poly}(B, D, \ell(t))$$

and polynomial working space, where $D = [K : \mathbb{Q}]$.

The algorithm also computes an integral basis of $\mathcal{O}_K$, its arithmetic and involution, and a suitable subfield chain within these bounds.

Proof. Compute the rational action matrix A of multiplication by θ in the supplied ideal basis. Since I is an $\mathcal{O}_K$-module, Lemma E.1 recovers the integral arithmetic and involution of $\mathcal{O}_K$ in polynomial time.

We then find a suitable subfield chain using Lemma 3.6(i) and apply Theorems 3.4 and 3.5 to I as a rank-one module.

The chain search takes $D^{O(\log D)}\,\mathrm{poly}(B, D)$ time and polynomial space, which is absorbed into the claimed bounds. □

F Auxiliary Algorithmic Reductions

We collect three auxiliary reductions used in Sections 4.5 and 5. The first converts exact SVP instances with rational Gram matrices into integer-basis instances without increasing the dimension. The second implements adaptive exact-SVP oracle reductions using randomized SVP algorithms while ensuring correctness of the complete computation. The third reduces enumeration of all closest vectors to repeated exact CVP calls, with polynomial additional space.

Lemma F.1. *Exact SVP for a positive definite rational $m \times m$ Gram matrix reduces in polynomial time to exact SVP on an integer square basis of dimension m, such that every shortest coefficient vector for the integer basis is shortest for the original Gram matrix.*

Proof. Let $G \in \mathbb{Q}^{m\times m}$ be positive definite, and choose a positive integer q such that qG has integer entries. Then all squared lengths $z^T G z$, for $z \in \mathbb{Z}^m$, belong to $q^{-1}\mathbb{Z}$. In particular, distinct squared lengths differ by at least $1/q$.

Let $H_0 = e_1^T G e_1 > 0$ and choose a positive rational number $\varepsilon < \min\left\{\frac{1}{4}, \frac{1}{10qH_0}\right\}$. Compute a nonsingular rational matrix C such that

$$(1-\varepsilon)x^T G x \le \|Cx\|^2 \le (1+\varepsilon)x^T G x \qquad \text{for all } x \in \mathbb{R}^m.$$

This can be done in polynomial time with polynomially many bits of precision. Indeed, rational positive definiteness gives a polynomial-bit lower bound on the smallest eigenvalue of G, so a sufficiently accurate rational approximation to a Cholesky factor satisfies the displayed inequalities.

Let $u \in \mathbb{Z}^m \setminus \{0\}$ minimize $u^T G u$, and let $v \in \mathbb{Z}^m \setminus \{0\}$ be any shortest coefficient vector for C. By the choice of v,

$$v^T G v \le \frac{v^T C^T C v}{1-\varepsilon} \le \frac{u^T C^T C u}{1-\varepsilon} \le \frac{1+\varepsilon}{1-\varepsilon} u^T G u.$$

Since $u^T G u \le H_0$, it follows that

$$0 \le v^T G v - u^T G u \le \frac{2\varepsilon}{1-\varepsilon} H_0 < \frac{1}{q}.$$

The squared lengths belong to $q^{-1}\mathbb{Z}$, so they must be equal. Thus every shortest coefficient vector for C is also shortest for G.

Finally, multiply C by a common positive denominator to obtain an integer nonsingular matrix A. This scaling does not change the shortest coefficient vectors. The entries of A have polynomial bit length, completing the reduction. □

We next justify replacing the exact oracle by a randomized SVP algorithm. The main issue is that successive oracle queries may depend on earlier answers, and a single incorrect answer may invalidate the complete computation.

Lemma F.2. *Consider a deterministic reduction on inputs of bit length β that makes at most Q_k exact SVP calls of dimension k, for every sequence of exact oracle answers. Let $Q = \sum_k Q_k \ge 1$.*

Suppose all such executions use polynomial working space, have intermediate bit lengths polynomial in β, and perform at most $Q\,\mathrm{poly}(\beta)$ work outside the oracle. Assume that the bounds Q_k and these resource bounds are computable in time polynomial in β, with $\log Q \le \mathrm{poly}(\beta)$.

Then the reduction has a randomized implementation whose complete output is correct with probability at least $2/3$, using polynomial working space and time

$$\mathrm{poly}(\beta)(1+\log Q)\left(Q + \sum_k Q_k\, k^{k/(4\mathrm{e})+o(k)}\right)$$

on every execution, including when the output is streamed.

In particular, if all oracle calls have dimension at most m, the running time is at most

$$\mathrm{poly}(\beta)\, Q(1+\log Q)\, m^{m/(4\mathrm{e})+o(m)}.$$

Proof. Consider an oracle query specified by a positive definite rational Gram matrix G of dimension k. By Lemma F.1, compute an integer basis A of the same dimension such that every shortest coefficient vector for A is shortest for G.

Run the randomized exact SVP algorithm of [FGL26, Theorem 4.1] independently

$$\eta = \lceil \log_3(3Q) \rceil$$

times. Each run returns a shortest nonzero vector with probability at least $2/3$, using polynomial space and time $\mathrm{poly}(|A|)k^{k/(4\mathrm{e})+o(k)}$.

For each returned vector v, compute $z = A^{-1}v$ and retain it if $z \in \mathbb{Z}^k \setminus \{0\}$. Among the retained vectors and a fixed nonzero fallback vector, return one minimizing $z^T G z$. If any run returns a shortest vector for A, the selected answer is shortest for G. Therefore, each query fails with probability at most $3^{-\eta}$.

Although the oracle queries are adaptive, the same failure bound holds conditionally on any history of exact earlier answers. Indeed, conditioned on such a history, the next query is fixed and uses fresh randomness. Thus the probability that the jth call is the first incorrect one is at most $3^{-\eta}$. By a union bound,

$$\Pr[\text{some incorrect oracle answer}] \le Q3^{-\eta} \le \frac{1}{3}.$$

Outside this event, the reduction follows an execution with exact oracle answers, and its complete output is correct.

To ensure the resource bounds on every execution, including those following an incorrect answer, the implementation monitors the number of oracle calls of each dimension, the non-oracle work, and the working space. It stops and reports failure before any prescribed bound is exceeded. These checks never interrupt an execution with exact oracle answers.

The repetitions are performed sequentially, storing only the current best candidate. Each dimension-k call therefore costs at most

$$\mathrm{poly}(\beta)(1+\log Q)\, k^{k/(4\mathrm{e})+o(k)}$$

time and polynomial space. Summing over the oracle calls and adding the non-oracle work proves the stated bounds. □

Lemma F.3. *Suppose exact CVP on an m-dimensional lattice L can be solved for every rational target. Then all closest vectors to any rational target t can be streamed without repetition using at most $2^m + 1$ CVP calls on L, with polynomial additional space.*

Proof. First compute $\delta^2 = \mathrm{dist}(t, L)^2$ using one CVP call. Two distinct closest vectors cannot belong to the same coset modulo $2L$. Indeed, if x, x' were distinct closest vectors with $x - x' \in 2L$, their midpoint would belong to L, whereas

$$\left\| \frac{x+x'}{2} - t \right\|^2 = \delta^2 - \frac{\|x-x'\|^2}{4} < \delta^2,$$

a contradiction.

Let B be a basis of L. Enumerate the 2^m coset representatives $y = Ba$ for $a \in \{0,1\}^m$. For each y, compute a closest vector $z_y \in L$ to $(t-y)/2$ and set $x_y = y + 2z_y$. Output x_y if and only if $\|x_y - t\|^2 = \delta^2$.

Each coset contains at most one globally closest vector, and the CVP call finds a closest point within that coset. Thus every closest vector is output exactly once. The procedure stores only one candidate at a time and uses $2^m + 1$ CVP calls. □

AI methodology

We used OpenAI Codex and ChatGPT, and Anthropic Claude, to propose and critique proofs, check assumptions and constants, compare claims with prior work, write verification code, and edit the manuscript. This assistance affected the algorithms, proofs and exposition throughout the paper, including the recursive enumeration along subfields and its arithmetic bounds, the search for a chain, and the use of the cited covering theorems for semistable lattices. The authors are responsible for the correctness and originality of all content, including references.

References

[ACK+21] Divesh Aggarwal, Yanlin Chen, Rajendra Kumar, Zeyong Li, and Noah Stephens-Davidowitz. Dimension-preserving reductions between SVP and CVP in different p-norms. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2444–2462, 2021. Full version: arXiv:2104.06576, version 1, 14 April 2021; section numbers refer to this version. doi:10.1137/1.9781611976465.145.

[ADRS15] Divesh Aggarwal, Daniel Dadush, Oded Regev, and Noah Stephens-Davidowitz. Solving the shortest vector problem in 2^n time using discrete Gaussian sampling. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC)*, pages 733–742, 2015. doi:10.1145/2746539.2746606.

[ADS15] Divesh Aggarwal, Daniel Dadush, and Noah Stephens-Davidowitz. Solving the closest vector problem in 2^n time—the discrete Gaussian strikes again! In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 563–582, 2015. doi:10.1109/FOCS.2015.41.

[AKS01] Miklós Ajtai, Ravi Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 601–610, 2001. doi:10.1145/380752.380857.

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of Mathematics*, 160(2):781–793, 2004. doi:10.4007/annals.2004.160.781.

[AS26] Martin R. Albrecht and Damien Stehlé. Module structure does not speed up enumeration. Private communication, 2026. Manuscript.

[Ban93] Wojciech Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296:625–635, 1993. doi:10.1007/BF01445125.

[BDK+18] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS – Kyber: A CCA-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 353–367. IEEE, 2018.

[BKLN26] Henry Bambury, Seungki Kim, Changmin Lee, and Phong Q. Nguyen. Adelic reduction of module lattices, 2026. Version 1, 26 September 2026. URL: https://arxiv.org/abs/2609.32936v1, arXiv:2609.32936.

[CDW17] Ronald Cramer, Léo Ducas, and Benjamin Wesolowski. Short Stickelberger class relations and application to ideal-SVP. In *Advances in Cryptology – EUROCRYPT*

2017, volume 10210 of *Lecture Notes in Computer Science*, pages 324–348. Springer, 2017. doi:10.1007/978-3-319-56620-7_12.

[CN18] Renaud Coulangeon and Gabriele Nebe. Slopes of Euclidean lattices, tensor product and group actions, 2018. Version 2, 23 November 2018. URL: https://arxiv.org/abs/1806.04984v2, arXiv:1806.04984.

[DEdP25] Léo Ducas, Lynn Engelberts, and Paola de Perthuis. Predicting module-lattice reduction. In *Advances in Cryptology – ASIACRYPT 2025*, Lecture Notes in Computer Science, pages 133–166. Springer, 2025. doi:10.1007/978-981-95-5099-9_5.

[DK16] Daniel Dadush and Gábor Kun. Lattice sparsification and the approximate closest vector problem. *Theory of Computing*, 12(2):1–34, 2016. doi:10.4086/toc.2016.v012a002.

[DKL+18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, 2018. doi:10.46586/tches.v2018.i1.238-268.

[dPT26] Paola de Perthuis and Filip Trenkić. Refined modelling of the primal attack, and variants against module-LWE. In *Post-Quantum Cryptography – PQCrypto 2026*, Lecture Notes in Computer Science, pages 342–376. Springer, 2026. doi:10.1007/978-3-032-22698-3_11.

[FGL26] Yansong Feng, Yiming Gao, and Jiaqi Liu. Faster SVP in polynomial space, 2026. Version 1, 18 September 2026. URL: https://arxiv.org/abs/2609.21612v1, arXiv:2609.21612.

[Gra84] Daniel R. Grayson. Reduction theory using semistability. *Commentarii Mathematici Helvetici*, 59:600–634, 1984. doi:10.1007/BF02566369.

[Hha26] Minki Hhan. Finding a shortest vector and more in $2^{n/2+o(n)}$ time using q-ary coset difference tree. Cryptology ePrint Archive, Paper 2026/1859, 2026. Received 2 September 2026; also arXiv:2609.02764v1. URL: https://eprint.iacr.org/2026/1859.

[HMR21] Farshid Hajir, Christian Maire, and Ravi Ramakrishna. Cutting towers of number fields. *Annales mathématiques du Québec*, 45(2):321–345, 2021. Full version: arXiv:1901.04354, version 1, 14 January 2019; section numbers refer to this version. doi:10.1007/s40316-021-00156-8.

[HS07] Guillaume Hanrot and Damien Stehlé. Improved analysis of Kannan's shortest lattice vector algorithm. In *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 170–186. Springer, 2007. Full version: arXiv:0705.0965, version 2, 9 May 2007; theorem numbers refer to this version. doi:10.1007/978-3-540-74143-5_10.

[Kan83] Ravi Kannan. Improved algorithms for integer programming and related lattice problems. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing (STOC)*, pages 193–206, 1983. doi:10.1145/800061.808749.

[KB79] Ravindran Kannan and Achim Bachem. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *SIAM Journal on Computing*, 8(4):499–507, 1979. doi:10.1137/0208040.

[KEF19] Paul Kirchner, Thomas Espitau, and Pierre-Alain Fouque. Algebraic and Euclidean lattices: Optimal lattice reduction and beyond. Cryptology ePrint Archive, Paper 2019/1436, 2019. URL: https://eprint.iacr.org/2019/1436.

[KEF20] Paul Kirchner, Thomas Espitau, and Pierre-Alain Fouque. Fast reduction of algebraic lattices over cyclotomic fields. In *Advances in Cryptology – CRYPTO 2020*, Lecture Notes in Computer Science, pages 155–185, 2020. doi:10.1007/978-3-030-56880-1_6.

[Len83] Arjen K. Lenstra. Factoring polynomials over algebraic number fields. In *Computer Algebra (EUROCAL 1983)*, volume 162 of *Lecture Notes in Computer Science*, pages 245–254, 1983. URL: https://ir.cwi.nl/pub/2541/2541D.pdf.

[LLL82] Arjen K. Lenstra, Hendrik W. Lenstra, Jr., and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982. doi:10.1007/BF01457454.

[LPMSW19] Changmin Lee, Alice Pellet-Mary, Damien Stehlé, and Alexandre Wallet. An LLL algorithm for module lattices. In *Advances in Cryptology – ASIACRYPT 2019*, volume 11922 of *Lecture Notes in Computer Science*, pages 59–90. Springer, 2019. doi:10.1007/978-3-030-34621-8_3.

[LRR26] Brayden Letwin, Victor Reis, and Thomas Rothvoss. Strongly log-concave measures and lattices, 2026. Version 1, 21 October 2026. URL: https://arxiv.org/abs/2610.29238v1, arXiv:2610.29238.

[MS20] Tamalika Mukherjee and Noah Stephens-Davidowitz. Lattice reduction for modules, or how to reduce ModuleSVP to ModuleSVP. In *Advances in Cryptology – CRYPTO 2020*, volume 12171 of *Lecture Notes in Computer Science*, pages 213–242. Springer, 2020. doi:10.1007/978-3-030-56880-1_8.

[MV13] Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. *SIAM Journal on Computing*, 42(3):1364–1391, 2013. Preliminary version in STOC 2010. doi:10.1137/100811970.

[PMHS19] Alice Pellet-Mary, Guillaume Hanrot, and Damien Stehlé. Approx-SVP in ideal lattices with pre-processing. In *Advances in Cryptology – EUROCRYPT 2019*, volume 11477 of *Lecture Notes in Computer Science*, pages 685–716. Springer, 2019. doi:10.1007/978-3-030-17656-3_24.

[PR07] Chris Peikert and Alon Rosen. Lattices that admit logarithmic worst-case to average-case connection factors. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 478–487, 2007. doi:10.1145/1250790.1250860.

[Rab80] Michael O. Rabin. Probabilistic algorithms in finite fields. *SIAM Journal on Computing*, 9(2):273–280, 1980. doi:10.1137/0209024.

[RS24] Oded Regev and Noah Stephens-Davidowitz. A reverse Minkowski theorem. *Annals of Mathematics*, 199(1):1–49, 2024. Full version: arXiv:1611.05979, version 6, 7 July 2022; section numbers refer to this version. doi:10.4007/annals.2024.199.1.1.

[TSZ14] Yuan Tian, Rongxin Sun, and Xueyong Zhu. A local-global approach to solving ideal lattice problems. In *Theory and Applications of Models of Computation (TAMC 2014)*, volume 8402 of *Lecture Notes in Computer Science*, pages 378–400, 2014. Preprint received 14 September 2013. URL: https://eprint.iacr.org/2013/594, doi:10.1007/978-3-319-06089-7_27.

[vKN13] Mark van Hoeij, Jürgen Klüners, and Andrew Novocin. Generating subfields. *Journal of Symbolic Computation*, 52:17–34, 2013. Preliminary version in ISSAC 2011; theorem and section numbers refer to the author manuscript of 3 June 2012. doi:10.1016/j.jsc.2012.05.010.