Twisted-PHS: Using the Product Formula to Solve Approx-SVP in Ideal Lattices

Olivier Bernard[1,2] and Adeline Roux-Langlois[1]

[1] Univ Rennes, CNRS, IRISA
{olivier.bernard, adeline.roux-langlois}@irisa.fr
[2] Thales, Gennevilliers, Laboratoire CHiffre

Abstract. Approx-SVP is a well-known hard problem on lattices, which asks to find short vectors on a given lattice, but its variant restricted to ideal lattices (which correspond to ideals of the ring of integers $\mathcal{O}_K$ of a number field K) is still not fully understood. For a long time, the best known algorithm to solve this problem on ideal lattices was the same as for arbitrary lattice. But recently, a series of works tends to show that solving this problem could be easier in ideal lattices than in arbitrary ones, in particular in the quantum setting.
Our main contribution is to propose a new "twisted" version of the PHS (by Pellet-Mary, Hanrot and Stehlé 2019) algorithm, that we call Twisted-PHS. As a minor contribution, we also propose several improvements of the PHS algorithm. On the theoretical side, we prove that our Twisted-PHS algorithm performs at least as well as the original PHS algorithm. On the practical side though, we provide a full implementation of our algorithm which suggests that much better approximation factors are achieved, and that the given lattice bases are a lot more orthogonal than the ones used in PHS. This is the first time to our knowledge that this type of algorithm is completely implemented and tested for fields of degrees up to 60.

Keywords: Ideal Lattices, Approx-SVP, PHS Algorithm

1 Introduction

Lattice-based cryptography is one of the most promising post-quantum solution to build cryptographic constructions, as shown by the large number of lattice-based submissions to the recent NIST post-quantum competition. Among those submissions, and the other recent more advanced constructions, several hard problems are used to build the security proofs, such as the Learning With Errors (LWE) problem [Reg05], its ring [SSTX09,LPR10] or module [LS15] variants (respectively Ring-LWE and Module-LWE) or the NTRU problem [HPS98]. In particular the Ring variant of the Learning With Errors problem is widely used as it seems to allow a nice trade-off between security and efficiency. Indeed, it is defined in a ring, usually $R = \mathbb{Z}/\langle x^n + 1\rangle$ for n a power of 2, whose structure allows constructions having a much better efficiency than if based on unstructured

problems like LWE. Concerning its hardness, there exists quantum worst-case to average case reductions [SSTX09,LPR10,PRS17] from the approx Shortest Vector Problem on ideal-lattices (Approx-id-SVP) to the Ring-LWE problem.

Approx-SVP is a well-known hard problem on lattices, which asks to find short vectors on a given lattice, but its variant restricted to ideal lattices (corresponding to ideals of the ring of integers R of a number field K) is still not fully understood. For a long time, the best known algorithm to solve this problem on ideal lattices was the same as for arbitrary lattices. The best trade-off in this case is given by Schnorr's hierarchy [Sch87], which allows to reach an approximation factor $2^{\tilde{O}(n^{\alpha})}$ in time $2^{\tilde{O}(n^{1-\alpha})}$, for $\alpha \in (0,1)$, using the BKZ algorithm. But recently, a series of works [CGS14,EHKS14,BS16,CDPR16,CDW17,DPW19,PHS19a] tends to show that solving this problem could be easier in ideal lattices than in arbitrary ones, in particular in the quantum setting.

Hardness of Approx-SVP on ideal lattices. This series of works starts with a claimed result [CGS14] of a quantum polynomial-time attack against a scheme named Soliloquy, which solves the Approx-SVP problem on a principal ideal lattice. The algorithm has two steps: the first one is solving the Principal Ideal Problem (PIP), and finds a generator of the ideal, the second one is solving a Closest-Vector Problem (CVP) in the log-unit lattice to find the shortest generator of the ideal. On one hand, the results of [EHKS14,BS16] on describing a quantum algorithm to compute class groups and then solve PIP in arbitrary degree number fields allow to have a quantum polynomial-time algorithm for the first step. On the other hand, a work by Cramer et al. [CDPR16] provides a full proof of the correctness of the algorithm described by [CGS14], and then concludes that there exists a polynomial-time quantum algorithm which solve Approx-SVP on ideal lattices for an approximation factor $2^{\tilde{O}(\sqrt{n})}$. In 2017, Cramer, Ducas and Wesolowski [CDW17] show how to use the Stickelberger lattice to generalize this result to any ideal lattice in prime power cyclotomic fields. The practical impact of their result was evaluated by the authors of [DPW19] by running extensive simulations. They conclude that the CDW algorithm should beat BKZ-300 for cyclotomic fields of degree larger than 24000.

In parallel, Pellet-Mary, Hanrot and Stehlé [PHS19a] proposed an extended version of [CDPR16,CDW17] which is now proven for any number fields K. The main feature of their algorithm, that we call PHS, is to use an exponential amount of preprocessing, depending only on K, in order to efficiently combine the two principal resolution steps of [CDW17], namely the CPMP (*Close Principal Multiple Problem*) and the SGP (*Shortest Generator Problem*). Combining these two steps in a single CVP instance provides some guarantee that the output of the CPMP solver has a generator which is "not much larger" than its shortest non-zero vector. Hence, the PHS algorithm in a number field K of degree n and discriminant Δ_K is split in two phases, given $\omega \in [0, 1/2]$:

1. The preprocessing phase builds a specific lattice, depending only on the field K, together with some hint allowing to efficiently solve Approx-CVP instances. This phase runs in time $2^{\tilde{O}(\log|\Delta_K|)}$ and outputs a hint $\mathcal{V}$ of bit-size $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)}$.

2. The query phase reduces each Approx-id-Svp challenge to an Approx-Cvp instance in this fixed lattice. It takes as inputs any ideal of $\mathcal{O}_K$, whose algebraic norm has bit-size bounded by $2^{\mathrm{poly}(\log|\Delta_K|)}$, the hint $\mathcal{V}$, and runs in time $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)} + \mathrm{T}_{\mathsf{Su}}(K)$. It outputs a non-zero element x of the ideal which solves Approx-Svp with an approximation factor $2^{\tilde{O}(\log^{\omega+1}|\Delta_K|/n)}$.

The term $\mathrm{T}_{\mathsf{Su}}(K)$ denotes the running time for computing S-unit groups which can then be used to compute class groups, unit groups, and class group discrete logarithms [BS16]. In the quantum world, $\mathrm{T}_{\mathsf{Su}}(K) = \tilde{O}\big(\ln|\Delta_K|\big)$ is polynomial, as shown in [BS16], building upon [EHKS14]. In the classical world, it remains subexponential in $\ln|\Delta_K|$, i.e. $\mathrm{T}_{\mathsf{Su}}(K) = \exp\tilde{O}(\ln^{\alpha}|\Delta_K|)$, where $\alpha = 1/2$ for prime power cyclotomic fields [BEF+17], and $\alpha = 2/3$ in the general case [BF14], being recently lowered to 3/5 by Gélin [Gél17].

Forgetting about the preprocessing cost, the query phase beats the traditional Schnorr's hierarchy [Sch87] when $\log|\Delta_K| \leq \tilde{O}(n^{1+\varepsilon})$ with $\varepsilon = 1/3$ in the quantum case, and $\varepsilon = 1/11$ in the classical case [PHS19a, Fig. 5.3]. It should be noted however that these bounds on the discriminant are not uniform as the approximation factor varies, e.g. for an approximation factor set to $2^{\sqrt{n}}$, the time complexity of the PHS algorithm asymptotically beats Schnorr's hierarchy only in the quantum case and only for $\varepsilon \leq 1/6$.

Our contribution. Our main contribution is to propose a new "twisted" version of the PHS [PHS19a] algorithm, that we call Twisted-PHS. As a minor contribution, we also propose several improvements of the PHS algorithm, in a optimized version described in §3.3. On the theoretical side, we prove that our Twisted-PHS algorithm performs at least as well as the original PHS algorithm, using the same Cvp solver using a preprocessing hint by Laarhoven [Laa16].

On the practical side though, we provide a full implementation of our algorithm, which suggests that much better approximation factors are achieved and that the given lattice bases are much more orthogonal than the ones used in [PHS19a]. To our knowledge, this is the first time that this type of algorithm is completely implemented and tested for fields of degrees up to 60. As a point of comparison, experiments of [PHS19a] constructed the log-S-unit lattice for cyclotomic fields of degrees at most 24, all but the last two being principal [PHS19a, Fig. 4.1]. We shall also mention the extensive simulations performed by [DPW19] using the Stickelberger lattice in prime power cyclotomic fields. Adapting these results to our construction is not immediate, as we need *explicit* S-units to compute our lattice. This is left for future work.

We explain our experiments in §5, where we evaluate three algorithms: the original PHS algorithm, as implemented in [PHS19b]; our optimized version Opt-PHS (§3.3), and our new twisted variant Tw-PHS (§4). We target two families of number fields, namely non-principal cyclotomic fields $\mathbb{Q}(\zeta_m)$ of prime conductors $m \in [\![23, 71]\!]$, and NTRU Prime fields $\mathbb{Q}(z_q)$ where z_q is a root of $x^q - x - 1$, for $q \in [\![23, 47]\!]$ prime. These correspond to the range of what is feasible in a reasonable amount of time in a classical setting. For cyclotomic fields, we managed to compute S-units up to $\mathbb{Q}(\zeta_{71})$ for all factor bases in less than a day, and

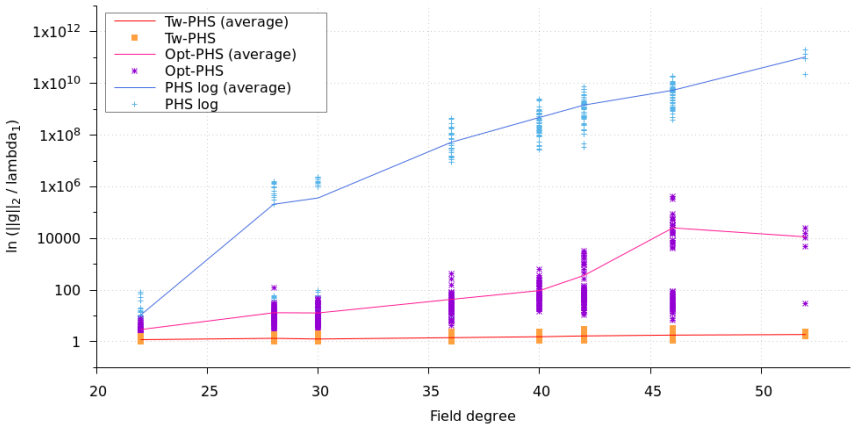

Fig. 1.1 – Approximation factors reached by Tw-PHS, Opt-PHS and PHS for cyclotomic fields of conductors 23, 29, 31, 37, 41, 43, 47 and 53 (in log scale).

all log-S-unit lattice variants up to $\mathbb{Q}(\zeta_{61})$. For NTRU Prime fields, we managed all computations up to $\mathbb{Q}(z_{47})$.

Experiments. We chose to perform three experiments to test the performances of our Twisted-PHS algorithm, and to compare it with the two other algorithms:

- We first evaluate the *geometric characteristics* of the lattice output by the preprocessing phase: the root Hermite factor δ_0, the orthogonality defect δ, and the average vector basis angle θ_{avg}, as described in details in §2.5. The last one seems difficult to interpret as it gives similar results in all cases, but the two other seem to show that the lattice output by Twisted-PHS is of better quality than in the two other cases. It shows significantly better root Hermite factor and orthogonality defect than any other lattice.
- For our second experiment, we evaluate *the Gram-Schmidt log norms* of each produced lattice. We propose two comparisons, the first one is before and after BKZ reduction to see the evolution of the norms in each case: it shows that the two curves are almost identical for Twisted-PHS but not for the other PHS variants. The second one is between the lattices output by the different algorithms, after BKZ reduction. The experiments emphasises that the decrease of the log norms seems much smaller in the twisted case than in the two other. Those two observations seem to corroborate the fact that the Twisted-PHS lattice is already quite orthogonal.
- Finally, we implemented all three algorithms from end to end and used them on numerous challenges to estimate their practically achieved *approximation factors*. This is to our knowledge the first time that these types of algorithms are completely run on concrete examples. The results of the experiments, shown in Fig. 1.1, suggest that the approximation factor reached by our algorithm increases very slowly with the dimension, in a way that could reveal subexponential or even better. We think that this last feature would be particularly interesting to prove.

Technical overview. We first quickly recall the principle of the PHS algorithm described in [PHS19a], which is split in two phases. The first phase consists in building a lattice that depends only on the number field K and allowing to express any Approx-id-Svp instance in K as an Approx-Cvp instance in the lattice. This preprocessing chooses a factor base FB, and builds an associated lattice consisting in the diagonal concatenation of some log-unit related lattice and the lattice of relations in the class group Cl_K between ideals of FB, with explicit generators. It then computes a hint of constrained size for the lattice to facilitate forthcoming Approx-Cvp queries. Concretely, they suggest to use Laarhoven's algorithm [Laa16], which for any $\omega \in [0, 1/2]$ outputs a hint $\mathcal{V}$ of bit-size bounded by $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)}$ that allows to deliver answers for approximation factors $\tilde{O}(\log|\Delta_K|^{\omega})$ in time bounded by the bit-size of $\mathcal{V}$ [Laa16, Cor. 1–2]. The second phase reduces the resolution of Approx-id-Svp to a single call to an Approx-Cvp oracle in the lattice output by the preprocessing phase, for any challenge ideal $\mathfrak{b}$ in the maximal order of K. The main idea of this reduction is to multiply the principal ideal output by the ClDL of $\mathfrak{b}$ on FB by ideals in FB until a "better" principal ideal is reached, i.e. having a short generator.

Our first contribution is to propose three improvements of the PHS algorithm. The first one consists in expliciting a candidate for the isometry used in the first preprocessing phase to build the lattice, and to use its geometric properties to derive a smaller lattice dimension, while still guaranteeing the same proven approximation factor. The last two respectively modify the composition of the factor base and the definition of the target vector in a way that significantly improves the approximation factor experimentally achieved by the second phase of the algorithm. Although these improvements do not modify the core of PHS algorithm and have no impact on the asymptotics, they nevertheless are of importance in practice, as shown by our experiments in §5.

We now explain our main contribution, called Twisted-PHS, which is based on the PHS algorithm. As in PHS algorithm, our algorithm relies on the so-called *log-S-unit lattice* with respect to a collection FB of prime ideals, called the factor base. This lattice captures local informations on FB, not only on (infinite) embeddings, to reduce a close principal multiple of a target ideal $\mathfrak{b}$ to a principal ideal containing $\mathfrak{b}$ which is guaranteed to have a somehow short generator. The main feature of our algorithm is to use the *Product Formula* to describe this log-S-unit lattice. This induces two major changes in PHS algorithm:

1. The first one is twisting the $\mathfrak{p}$-adic valuations by $\ln \mathcal{N}(\mathfrak{p})$, giving weight to the fact that using a relation increasing the valuations at big norm ideals costs more than a relation involving smaller norm ideals.
2. The second one is projecting the target directly inside the log-S-unit lattice and not only into the unit log-lattice corresponding to fundamental units.

In fact, the way our twisted version uses S-units with respect to FB to reduce the solution of the ClDL problem can be viewed as a natural generalization of the way classical algorithms reduce principal ideal generators using regular units.

Adding weights $\ln \mathcal{N}(\mathfrak{p})$ to integer valuations at any prime ideal $\mathfrak{p}$ intuitively allows to make a more relevant combination of the S-units we use to reduce the

output of the ClDL, quantifying the fact that increasing valuations at big norm prime ideals costs more than increasing valuations at small norm prime ideals. Besides, the product formula induces the possibility to project elements on the whole log-S-unit lattice instead of projecting only on the subspace corresponding to the log-unit lattice. As a consequence, it maintains inside the lattice the size and the algebraic norm logarithm of the S-units. At the end, the CVP solver in this alternative lattice combines more efficiently the goal of minimizing the algebraic norm for the CPMP while still guaranteeing a small size for the SGP solution in the obtained principal multiple.

In §4, we describe two versions of our Twisted-PHS algorithm. The first one, composed by $\mathcal{A}^{(\mathrm{Laa})}_{\mathsf{tw\text{-}pcmp}}$ and $\mathcal{A}^{(\mathrm{Laa})}_{\mathsf{tw\text{-}query}}$ is proven to perform at least as well as the original PHS algorithm with the same CVP solver using a preprocessing hint by Laarhoven. But in practice, we propose two alternative algorithms $\mathcal{A}^{(\mathrm{bkz})}_{\mathsf{tw\text{-}pcmp}}$ and $\mathcal{A}^{(\mathrm{np})}_{\mathsf{tw\text{-}query}}$ with the following differences. Algorithm $\mathcal{A}^{(\mathrm{bkz})}_{\mathsf{tw\text{-}pcmp}}$ performs a minimal reduction step of the lattice as sole lattice preprocessing to smooth the input basis. Algorithm $\mathcal{A}^{(\mathrm{np})}_{\mathsf{tw\text{-}query}}$ resorts to Babai's Nearest Plane algorithm for the CVP solver role. Experimental evidence in §5 suggest that these algorithms perform remarkably well, because the twisted description of the log-S-unit lattice seems much more orthogonal than expected. Proving this property would remove, in a quantum setting, the only part that is not polynomial in $\ln|\Delta_K|$.

2 Preliminaries

Notations. A vector is designated by a bold letter $\mathbf{v}$, its i-th coordinate by v_i and its ℓ_p-norm, $p \in \mathbb{N}^* \cup \{\infty\}$, by $\|\mathbf{v}\|_p$. As a special case, the n-dimensional vector whose coefficients are all 1's is written $\mathbf{1}_n$. All matrices will be given using *row* vectors, $\mathcal{D}_{\mathbf{v}}$ is the diagonal matrix with coefficients v_i on the diagonal, I_n is the identity and $\mathbf{1}_{n\times n}$ denotes the square matrix of dimension n filled with 1's.

2.1 Number fields, ideals and class groups

In this paper, K always denotes a number field of degree n over $\mathbb{Q}$ and $\mathcal{O}_K$ its maximal order. The algebraic trace and norm of $\alpha \in K$, resp. denoted by $\mathrm{Tr}(\alpha)$ and $\mathcal{N}(\alpha)$, are defined as the trace and determinant of the endomorphism $x \mapsto \alpha x$ of K, viewed as a $\mathbb{Q}$-vector space. The discriminant of K is written Δ_K and can be defined, for any $\mathbb{Z}$-basis $\omega_1, \ldots, \omega_n$ of $\mathcal{O}_K$, as $\det\big(\mathrm{Tr}(\omega_i\omega_j)\big)_{i,j}$. Most complexities of number theoretic algorithms depend on $\ln|\Delta_K|$.

The fractional ideals of K are designated by gothic letters, like $\mathfrak{b}$, and form a multiplicative group $\mathcal{I}_K$. The class group Cl_K of K is the quotient group of $\mathcal{I}_K$ with its subgroup of principal ideals $\mathcal{P}_K \stackrel{\text{def}}{:=} \big\{\langle\alpha\rangle, \text{ for all } \alpha \in K\big\}$. The class group is a finite group, whose order h_K is called the class number of K. For any ideal $\mathfrak{b} \in \mathcal{I}_K$, the class of $\mathfrak{b}$ in Cl_K is denoted by $\big[\mathfrak{b}\big]$.

We will specifically target two families of number fields, widely used in cryptography [Pei16]: cyclotomic fields $\mathbb{Q}(\zeta_m)$, where ζ_m is a primitive m-th

root of unity, and NTRU Prime [BCLV17] fields $\mathbb{Q}(z_q)$, where z_q is a root of $x^q - x - 1$ for q prime. Both families have discriminants of order n^n. More exactly, for cyclotomic fields $\mathcal{O}_{\mathbb{Q}(\zeta_m)} = \mathbb{Z}[\zeta_m]$, so we have [Was97, Pr. 2.7]: $\Delta_{\mathbb{Q}(\zeta_m)} = (-1)^{\varphi(m)/2} \frac{m^{\varphi(m)}}{\prod_{p|m} p^{\varphi(m)/(p-1)}}$.

For NTRU Prime fields, the siuation is marginally more involved, as $\mathbb{Z}[z_q]$ is maximal if and only if its discriminant $D_0 = q^q - (q-1)^{q-1}$ [Swa62, Th. 2] is squarefree [Kom75, Th. 4]: $\Delta_{\mathbb{Q}(z_q)} = \prod_{p|D_0} p^{v_p(D_0) \bmod 2}$, where $p^{v_p(D_0)}$ divides exactly D_0. Note however that there is strong evidence that such D_0's are generically squarefree, say with probability roughly 0.99 [BMT15, Conj. 1.1]. Actually, we checked that the conductor of $\mathbb{Z}[z_q]$ is not divisible by any of the first 10^6 primes for all $q \leq 1000$ outside the set $\{257, 487\}$, for which $59^2 \mid D_0$.

2.2 The product formula

Let (r_1, r_2) be the signature of K with $n = r_1 + 2r_2$. The real embeddings of K are numbered from σ_1 to σ_{r_1}, whereas the complex embeddings come in pairs $\left(\sigma_j, \overline{\sigma}_j\right)$ for $j \in [\![r_1 + 1, r_2]\!]$.

Each embedding σ of K into $\mathbb{C}$ induces an archimedean absolute value $|\cdot|_\sigma$ on K, such that for $\alpha \in K$, $|\alpha|_\sigma = |\sigma(\alpha)|$; two complex conjugate embeddings yield the same absolute value. Thus, it is common to identify the set $\mathcal{S}_\infty$ of infinite places of K with the embeddings of K into $\mathbb{C}$ up to conjugation, so that $\mathcal{S}_\infty = \left\{\sigma_1, \ldots, \sigma_{r_1}, \sigma_{r_1+1}, \ldots, \sigma_{r_1+r_2}\right\}$. The completion of K with respect to the absolute value induced by an infinite place $\sigma \in \mathcal{S}_\infty$ is denoted by K_σ; it is $\mathbb{R}$ (resp. $\mathbb{C}$) for real places (resp. complex places).

Likewise, let $\mathfrak{p}$ be a prime ideal of $\mathcal{O}_K$ above $p \in \mathbb{Z}$ of residue degree f. For $\alpha \in K$, the largest power of $\mathfrak{p}$ that divides $\langle \alpha \rangle$ is called the valuation of α at $\mathfrak{p}$, and denoted by $v_\mathfrak{p}(\alpha)$; this defines a non-archimedean absolute value $|\cdot|_\mathfrak{p}$ on K such that $|\alpha|_\mathfrak{p} = p^{-v_\mathfrak{p}(\alpha)}$. This absolute value can also be viewed as induced by any of the f embeddings of K into its $\mathfrak{p}$-adic completion $K_\mathfrak{p} \subseteq \mathbb{C}_p$, which is an extension of $\mathbb{Q}_p$ of degree f. Hence, the set $\mathcal{S}_0$ of finite places of K is specified by the infinite set of prime ideals of $\mathcal{O}_K$, and Ostrowski's theorem for number fields ([Con, Th. 3], [Nar04, Th. 3.3]) states that all non archimedean absolute values on K are obtained in this way, up to equivalence.

Probably the most interesting thing is that these absolute values are tied together by the following product formula ([Con, Th. 4], [Nar04, Th. 3.5]):

$$\prod_{\sigma \in \mathcal{S}_\infty} |\alpha|_\sigma^{[K_\sigma : \mathbb{R}]} \cdot \prod_{\mathfrak{p} \in \mathcal{S}_0 \supset p\mathbb{Z}} |\alpha|_\mathfrak{p}^{[K_\mathfrak{p} : \mathbb{Q}_p]} \left(= \mathcal{N}(\alpha) \cdot \prod_{\mathfrak{p} \in \mathcal{S}_0} \mathcal{N}(\mathfrak{p})^{-v_\mathfrak{p}(\alpha)} \right) = 1. \tag{2.1}$$

As all but finitely many of the $|\alpha|_v$'s, for $v \in \mathcal{S}_\infty \cup \mathcal{S}_0$, are 1, their product is really a finite product. Note that the $\mathcal{S}_\infty$ part is $|\mathcal{N}(\alpha)|$, and each term of the $\mathcal{S}_0$ part can be written as $\mathcal{N}(\mathfrak{p})^{-v_\mathfrak{p}(\alpha)}$. This formula is actually a natural generalization to number fields of the innocuous looking product formula for $r \in \mathbb{Q}$, written as: $|r| \cdot \prod_{p \text{ prime}} p^{-v_p(r)} = 1$.

2.3 Unit groups

A more thorough version of this section is given in the full version [BR20, §2.3]. Let $\mathcal{O}_K^\times$ be the multiplicative group of units of $\mathcal{O}_K$, i.e. the group of all elements of K of algebraic norm ± 1, and let $\mu(\mathcal{O}_K^\times)$ be its torsion subgroup of roots of unity of K. Classically, the logarithmic embedding from K to $\mathbb{R}^{r_1+r_2}$ is defined as [Coh93, Def. 4.9.6]: $\mathrm{Log}_\infty\, \alpha = ([K_\sigma : \mathbb{R}] \cdot \ln|\sigma(\alpha)|)_{\sigma \in \mathcal{S}_\infty}$. Actually, it will be more convenient to use a *flat* logarithmic embedding from K to $\mathbb{R}^{r_1+2r_2}$, as in [PHS19a,BDPW20], and defined as follows:

$$\overline{\mathrm{Log}}_\infty\, \alpha = \Big(\big\{\ln|\sigma_i(\alpha)|\big\}_{i\in[\![1,r_1]\!]}, \big\{\ln|\sigma_{r_1+j}(\alpha)|, \ln|\overline{\sigma}_{r_1+j}(\alpha)|\big\}_{j\in[\![1,r_2]\!]}\Big). \tag{2.2}$$

Dirichlet's unit theorem [Nar04, Th. 3.13] states that $\mathcal{O}_K^\times$ is a finitely generated abelian group of rank $\nu = r_1+r_2-1$. Further, its image $\overline{\mathrm{Log}}_\infty\, \mathcal{O}_K^\times$ under the flat logarithmic embedding is a lattice, called the *log-unit lattice*, which spans H_0, defined as $\mathcal{L}_0 \cap \mathbb{R}_0^n$, i.e. the intersection of the trace zero hyperplane of $\mathbb{R}^n$ and of $\mathcal{L}_0 = \big\{\mathbf{y} \in \mathbb{R}^n : y_{r_1+2j-1} = y_{r_1+2j}, j \in [\![1, r_2]\!]\big\}$: there exist fundamental torsion-free elements $\varepsilon_1, \ldots, \varepsilon_\nu \in \mathcal{O}_K^\times$ such that:

$$\mathcal{O}_K^\times \simeq \mu(\mathcal{O}_K^\times) \times \varepsilon_1^{\mathbb{Z}} \times \cdots \times \varepsilon_\nu^{\mathbb{Z}}. \tag{2.3}$$

Let $\Lambda_K = (\mathrm{Log}_\infty\, \varepsilon_i)_{1\le i\le \nu}$ be any $\mathbb{Z}$-basis of $\mathrm{Log}_\infty\, \mathcal{O}_K^\times$. The regulator of K, written R_K, quantifies the density of the unit group in K. It is defined as the absolute value of the determinant of $\Lambda_K^{(j)}$, where $\Lambda_K^{(j)}$ is the submatrix of Λ_K without the j-th coordinate, for any $j \in [\![1, r_1+r_2]\!]$.

On the S-unit group. The S-unit group generalizes the unit group $\mathcal{O}_K^\times$ by allowing inverses of elements whose valuations are non zero exactly over a chosen finite set of primes of $\mathcal{S}_0$. Let $\mathrm{FB} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_k\}$ be such a factor basis, and let $\mathcal{O}_{K,\mathrm{FB}}^\times$ denote the S-unit group of K with respect to FB. Formally, we have $\mathcal{O}_{K,\mathrm{FB}}^\times = \big\{\alpha \in K : \exists e_1, \ldots, e_k \in \mathbb{Z}, \langle\alpha\rangle = \prod \mathfrak{p}_j^{e_j}\big\}$. Similarly, we define a flat S-logarithmic embedding [Nar04, §3, p.98] from K to $\mathcal{L} = \mathcal{L}_0 \times \mathbb{R}^k$ by:

$$\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\, \alpha = \Big(\overline{\mathrm{Log}}_\infty\, \alpha, \big\{-v_\mathfrak{p}(\alpha) \cdot \ln \mathcal{N}(\mathfrak{p})\big\}_{\mathfrak{p}\in\mathrm{FB}}\Big). \tag{2.4}$$

From the product formula (2.1), the image of $\mathcal{O}_{K,\mathrm{FB}}^\times$ lies in $H = \mathcal{L} \cap \mathbb{R}_0^{n+k}$, the trace zero hyperplane of $\mathcal{L}$. This fact is used to prove the following theorem:

Theorem 2.1 (Dirichlet-Chevalley-Hasse [Nar04, Th. III.3.12]). *The S-unit group is a finitely generated abelian group of rank $\sharp\mathcal{S}_\infty + \sharp\mathrm{FB} - 1$. Further, the image $\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\big(\mathcal{O}_{K,\mathrm{FB}}^\times / \mu(\mathcal{O}_K^\times)\big)$ is a lattice which spans the $(\nu+k)$-dimensional space H: there exist fundamental torsion-free S-units $\eta_1, \ldots, \eta_k \in \mathcal{O}_{K,\mathrm{FB}}^\times$ st.:*

$$\mathcal{O}_{K,\mathrm{FB}}^\times \simeq \mu(\mathcal{O}_K^\times) \times \varepsilon_1^{\mathbb{Z}} \times \cdots \times \varepsilon_\nu^{\mathbb{Z}} \times \eta_1^{\mathbb{Z}} \times \cdots \times \eta_k^{\mathbb{Z}}.$$

Let $\widetilde{\Lambda}_{K,\mathrm{FB}} = \left(\{\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\varepsilon_i\}, \{\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\eta_j\}\right)$ be a row basis of $\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\mathcal{O}_{K,\mathrm{FB}}^{\times}$, which will be called the *log-S-unit lattice*. Using that $\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\varepsilon_i$ is uniformly zero on coordinates corresponding to finite places, the shape of $\widetilde{\Lambda}_{K,\mathrm{FB}}$ is:

$$\widetilde{\Lambda}_{K,\mathrm{FB}} \stackrel{\text{def}}{:=} \left[\begin{array}{c|c} \widetilde{\Lambda}_K & 0 \\ \hline \overline{\mathrm{Log}}_{\infty}\,\eta_1 & \\ \vdots & \left(-v_{\mathfrak{p}_j}(\eta_i)\ln\mathcal{N}(\mathfrak{p}_j)\right)_{1\le i,j\le k} \\ \mathrm{Log}_{\infty}\,\eta_k & \end{array}\right]. \tag{2.5}$$

Similarly, Th. 2.1 allows to define the S-regulator $R_{K,\mathrm{FB}}$ of K wrpt. FB as the absolute value of any of the (r_1+r_2+k) minors of any row basis $\Lambda_{K,\mathrm{FB}}$ of $\mathrm{Log}_{\infty,\mathrm{FB}}\,\mathcal{O}_{K,\mathrm{FB}}^{\times}$. The value of $R_{K,\mathrm{FB}}$ is given by the following proposition:

Proposition 2.2. *Let $h_K^{(\mathrm{FB})}$ the cardinal of the subgroup $\mathrm{Cl}_K^{(\mathrm{FB})}$ of Cl_K generated by classes of ideals in* FB*. Then, the S-regulator $R_{K,\mathrm{FB}}$ can be written as: $R_{K,\mathrm{FB}} = h_K^{(\mathrm{FB})} R_K \prod_{\mathfrak{p}\in\mathrm{FB}} \ln\mathcal{N}(\mathfrak{p})$.*

The proof is given in the full version. We stress that the S-regulator could not be consistently defined anymore if these twistings by the $\ln\mathcal{N}(\mathfrak{p})$'s were removed, as in this case, the property that all columns sum to 0 disappears. Finally, the volume of the log-S-unit lattice is tied to $R_{K,\mathrm{FB}}$ by the following proposition, which generalizes [BDPW20, Lem. A.1], and that we also prove in [BR20]:

Proposition 2.3. *Under the flat S-logarithmic embedding, the log-S-unit lattice has volume: $\mathrm{Vol}\left(\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\mathcal{O}_{K,\mathrm{FB}}^{\times}\right) = \sqrt{n+k}\cdot 2^{-r_2/2}\cdot h_K^{(\mathrm{FB})} R_K \prod_{\mathfrak{p}\in\mathrm{FB}}\ln\mathcal{N}(\mathfrak{p})$. Using an empty factor basis, it implies $\mathrm{Vol}\left(\overline{\mathrm{Log}}_{\infty}\,\mathcal{O}_K^{\times}\right) = \sqrt{n}\cdot 2^{-r_2/2}\cdot R_K$.*

2.4 Algorithmic number theory

This section is split into §2.4 and §2.5 in the full version [BR20]. The former recalls useful number theoretic bounds and relations, such as the *analytic class number formula*, allowing to bound $h_K R_K$, Bach's bound on the algebraic norm of class group generators, and the *Prime Ideal Theorem* on the density of prime ideals. All rely on the *Generalized Riemann Hypothesis* (GRH). We only recall problem definitions discussed in the latter, the most essential being the ClDL.

Problem 2.4 (Class Group Discrete Logarithm (ClDL) [BS16]). Given a set FB of prime ideals generating a subgroup $\mathrm{Cl}_K^{(\mathrm{FB})}$ of Cl_K, and a fractional ideal $\mathfrak{b}$ st. $\left[\mathfrak{b}\right] \in \mathrm{Cl}_K^{(\mathrm{FB})}$, output $\alpha\in K$ and $v_i\in\mathbb{Z}$ st. $\langle\alpha\rangle = \mathfrak{b}\cdot\prod_{\mathfrak{p}_i\in\mathrm{FB}}\mathfrak{p}_i^{v_i}$.

Problem 2.5 (Close Principal Multiple Problem (CPMP) [CDW17, §2.2]). Given a fractional ideal $\mathfrak{b}$, output a "reasonably small" integral ideal $\mathfrak{c}$ such that $\left[\mathfrak{c}\right] = \left[\mathfrak{b}\right]^{-1}$.

Problem 2.6 (Shortest Generator Problem (SGP)). Given $\mathfrak{a} = \langle\alpha\rangle$, principal ideal generated by some $\alpha\in K$, find the shortest $\alpha'\in\mathfrak{a}$ such that $\mathfrak{a} = \langle\alpha'\rangle$.

2.5 Lattices geometry and hard problems

Let L be a lattice. For any $p \in \mathbb{N}^* \cup \{\infty\}$ and $1 \leq i \leq \dim L$, the i-th minimum $\lambda_i^{(p)}(L)$ of L for the ℓ_p-norm is the minimum radius $r > 0$ such that $\{\mathbf{v} \in L : \|\mathbf{v}\|_p \leq r\}$ has rank i [NV10, Def. 2.13]. For any $\mathbf{t}$ in the span of L, the distance between $\mathbf{t}$ and L is $\mathrm{dist}_p(\mathbf{t}, L) = \inf_{\mathbf{v}\in L}\|\mathbf{t} - \mathbf{v}\|_p$, and the *covering radius* of L wrpt. ℓ_p-norm is $\mu_p(L) = \sup_{\mathbf{t}\in L\otimes\mathbb{R}} \mathrm{dist}_p(\mathbf{t}, L)$. For the euclidean norm, we omit $p = 2$ most of the time.

A fractional ideal $\mathfrak{b}$ of K can be seen, under the canonical embedding, as a full rank lattice in $\mathbb{R}^n$, called an *ideal lattice*, of volume $\sqrt{|\Delta_K|} \cdot \mathcal{N}(\mathfrak{b})$. The arithmetic-geometric mean inequality, using that $|\mathcal{N}(\alpha)| \geq \mathcal{N}(\mathfrak{b})$ for all $\alpha \in \mathfrak{b}$, and the Minkowski's inequality [NV10, Th. 2.4] imply:

$$\mathcal{N}(\mathfrak{b})^{1/n} \leq \lambda_1^{(\infty)}(\mathfrak{b}) \leq \sqrt{|\Delta_K|}^{1/n}\mathcal{N}(\mathfrak{b})^{1/n} \tag{2.6}$$

$$\sqrt{n} \cdot \mathcal{N}(\mathfrak{b})^{1/n} \leq \lambda_1^{(2)}(\mathfrak{b}) \leq \sqrt{n} \cdot \sqrt{|\Delta_K|}^{1/n}\mathcal{N}(\mathfrak{b})^{1/n} \tag{2.7}$$

More precisely, $\lambda_1(\mathfrak{b}) \leq (1+o(1))\sqrt{2n/\pi e}\cdot\mathrm{Vol}^{1/n}(\mathfrak{b})$, and the Gaussian Heuristic for full rank random lattices [NV10, Def. 2.8] predicts $\lambda_1(\mathfrak{b}) \approx \sqrt{n/2\pi e}\cdot\mathrm{Vol}^{1/n}(\mathfrak{b})$ on average. In the case of ideal lattices, this yields a pretty good estimation of the shortness of vectors, even if $\lambda_1(\mathfrak{b})$ is not known precisely.

We will consider the following algorithmic lattice problems. Both problems can be readily restricted to ideal lattices under the labels Approx-id-Svp and Approx-id-Cvp.

Problem 2.7 (Approximate Shortest Vector Problem (Approx-SVP) [NV10, Pb. 2.2]). Given a lattice L and an approximation factor $\gamma \geq 1$, find a vector $\mathbf{v} \in L$ such that $\|\mathbf{v}\| \leq \gamma \cdot \lambda_1(L)$.

Problem 2.8 (Approximate Closest Vector Problem (Approx-CVP) [NV10, Pb. 2.5]). Given a lattice L, a target $\mathbf{t} \in L \otimes \mathbb{R}$ and an approximation factor $\gamma \geq 1$, find a vector $\mathbf{v} \in L$ such that $\|\mathbf{t} - \mathbf{v}\| \leq \gamma \cdot \mathrm{dist}(\mathbf{t}, L)$.

Actually, it will be more convenient to work with a slightly modified version of Approx-Cvp, where the output is required to be at distance absolutely bounded by some B, independently of the target distance to the lattice. By abuse of terminology, we still call this variant Approx-Cvp.

Evaluating the quality of a lattice basis. Let $B = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$ be a basis of a full rank n-dimensional lattice L, and let the Gram-Schmidt Orthogonalization of B be $\mathrm{GSO}(B) = (\mathbf{b}_1^\star, \ldots, \mathbf{b}_n^\star)$. Approximation algorithms usually attempt to compute a *good* basis of the given lattice, i.e. whose vectors are as short and as orthogonal as possible. These lattice reduction algorithms, such as LLL [LLL82] or BKZ [CN11], try to limit the decrease of the Gram-Schmidt norms $\|\mathbf{b}_i^\star\|$: intuitively, a wide gap in this sequence reveals that $\mathbf{b}_i$ is far from orthogonal to $\langle \mathbf{b}_1, \ldots, \mathbf{b}_{i-1} \rangle$. Evaluating the quality of a lattice basis is actually a tricky task that depends partly on the targeted problem (see e.g. [Xu13]). We will use the following geometric metrics:

1. the root-Hermite factor δ_0 is widely used to measure the performance of lattice reduction algorithms [NS06,GN08,CN11], especially for solving Svp-like problems: $\delta_0^n(B) = \frac{\|\mathbf{b}_1\|}{\mathrm{Vol}^{1/n} L}$. Experimental evidence suggest that on average, LLL achieves $\delta_0^{\mathsf{LLL}} \approx 1.02$ [NS06,GN08] and BKZ with block size b achieves $\delta_0^{\mathsf{BKZ}_b} \approx \left(\frac{b}{2\pi e}(\pi b)^{1/b}\right)^{1/(2b-2)}$ for $b \geq 50$ [Che13,CN11].
2. the normalized orthogonality defect δ [MG02, Def. 7.5] captures the global quality of the basis, not just of the first vector, and is especially useful for Cvp-like problems e.g. if the lattice possesses abnormally short vectors: $\delta^n(B) = \frac{\prod_{i=1}^n \|\mathbf{b}_i\|}{\mathrm{Vol}\, L}$. For purely orthogonal bases $\delta = 1$, and its smallest possible value is $\left(\prod_i \lambda_i(L)/\mathrm{Vol}\, L\right)^{1/n} \leq \sqrt{1+\frac{n}{4}}$ by Minkowski's second theorem [NV10, Th. 2.5].
3. the minimum vector basis angle, defined as [Xu13, Eq. (15)]: $\theta_{\mathsf{min}}(B) = \min_{1\leq i<j\leq n} \min\{\theta_{ij}, \pi - \theta_{ij}\}$ for $\theta_{ij} = \frac{\arccos\langle \mathbf{b}_i, \mathbf{b}_j\rangle}{\|\mathbf{b}_i\|\|\mathbf{b}_j\|}$. We propose to consider the mean vector basis angle $\theta_{\mathsf{avg}}(B)$, which averages over all $\min\{\theta_{ij}, \pi - \theta_{ij}\}$.

3 The PHS algorithm

This section describes the PHS algorithm for solving Approx-id-Svp, as introduced by Pellet-Mary, Hanrot and Stehlé in [PHS19a], and discusses several improvements. The PHS algorithm extends the techniques from [CDPR16,CDW17] to any number field K and is split in two phases:

1. the preprocessing phase $\mathcal{A}_{\mathsf{pre\text{-}proc}}$, described in §3.1, builds a specific lattice together with some hint allowing to efficiently solve Approx-Cvp instances;
2. the query phase $\mathcal{A}_{\mathsf{query}}$, detailed in §3.2, reduces each Approx-id-Svp challenge to an Approx-Cvp instance in this fixed lattice.

More precisely, under the GRH and several heuristic assumptions detailed in [PHS19a, H. 1–6], they prove the following theorem:

Theorem 3.1 ([PHS19a, Th. 1.1]). *Let $\omega \in [0, 1/2]$ and K be a number field of degree n and discriminant Δ_K with a known basis of $\mathcal{O}_K$. Under some conjectures and heuristics, there exist two algorithms $\mathcal{A}_{\mathsf{pre\text{-}proc}}$ and $\mathcal{A}_{\mathsf{query}}$ such that:*

- *Algorithm $\mathcal{A}_{\mathsf{pre\text{-}proc}}$ takes as input $\mathcal{O}_K$, runs in time $2^{\tilde{O}(\log|\Delta_K|)}$ and outputs a hint $\mathcal{V}$ of bit-size $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)}$;*
- *Algorithm $\mathcal{A}_{\mathsf{query}}$ takes as inputs any ideal $\mathfrak{b}$ of $\mathcal{O}_K$, whose algebraic norm has bit-size bounded by $2^{\mathrm{poly}(\log|\Delta_K|)}$, and the hint $\mathcal{V}$ output by $\mathcal{A}_{\mathsf{pre\text{-}proc}}$, runs in time $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)} + \mathrm{T}_{Su}(K)$, and outputs a non-zero element $x \in \mathfrak{b}$ such that $\|x\|_2 \leq 2^{\tilde{O}(\log^{\omega+1}|\Delta_K|/n)} \cdot \lambda_1(\mathfrak{b})$.*

We start by describing the preprocessing phase $\mathcal{A}_{\mathsf{pre\text{-}proc}}$ in §3.1, then the query phase together in §3.2. We thereafter discuss several algorithmic and theoretic minor improvements in §3.3.

3.1 Preprocessing of the number field

From a number field K and a size parameter $\omega \in [0, 1/2]$, the preprocessing phase consists in building and preparing a lattice L_{phs} that depends only on the number field K and allows to express any Approx-id-SVP instance in K as an Approx-CVP instance in L_{phs}. The most significant part of this preprocessing is devoted to the computation of a hint of constrained size that can be used to facilitate those forthcoming Approx-CVP queries.

We first define the lattice which is used in [PHS19a], discuss how the authors derive its dimension from volume considerations, and then expose the full preprocessing algorithm.

Definition of the lattice L_{phs}. Let $\mathrm{FB} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_k\}$ be a set of prime ideals generating the class group Cl_K. The lattice L_{phs} proposed in [PHS19a, §3.1] consists in the diagonal concatenation of some log-unit related lattice and the lattice of relations in Cl_K between ideals of FB, with explicit generators. Formally, it is generated by the $(\nu + k)$ rows of the following square matrix:

$$B_{L_{\mathsf{phs}}} \stackrel{\text{def}}{:=} \left[\begin{array}{c|c} c \cdot B_\Lambda & 0 \\ \hline c \cdot f_{H_0}(\mathbf{h}^{(0)}_{\eta_1}) & \\ \vdots & \ker \mathfrak{f}_{\mathrm{FB}} = \left(-v_{\mathfrak{p}_j}(\eta_i)\right)_{1 \le i,j \le k} \\ c \cdot f_{H_0}(\mathbf{h}^{(0)}_{\eta_k}) & \end{array}\right], \tag{3.1}$$

- where f_{H_0} is an isometry from $H_0 \subset \mathbb{R}^n$ to $\mathbb{R}^\nu$, where H_0 is the intersection of the span $\mathcal{L}_0$ of $\overline{\mathrm{Log}}_\infty \mathcal{O}_K$, i.e. $\mathcal{L}_0 = \{\mathbf{y} \in \mathbb{R}^n : y_{r_1+2i-1} = y_{r_1+2i}, i \in [\![1, r_2]\!]\}$, and of the trace zero hyperplane $\mathbb{R}^n_0 = \mathbf{1}_n^\perp$;
- the matrix B_Λ is a row basis of $f_{H_0}(\overline{\mathrm{Log}}_\infty \mathcal{O}_K^\times)$;
- the bottom right part of $B_{L_{\mathsf{phs}}}$ generates the lattice of all relations in Cl_K between ideals of FB, i.e. is the kernel of $\mathfrak{f}_{\mathrm{FB}} : (e_1, \ldots, e_k) \in \mathbb{Z}^k \mapsto \prod_j [\mathfrak{p}_j]^{e_j}$;
- each row basis vector $\mathbf{v}_i = (v_{i1}, \ldots, v_{ik})$ of $\ker \mathfrak{f}_{\mathrm{FB}}$ is associated to $\eta_i \in K$ such that $\langle \eta_i \rangle \cdot \prod_j \mathfrak{p}_j^{v_{ij}} = \mathcal{O}_K$, thus $v_{ij} = -v_{\mathfrak{p}_j}(\eta_i)$, and $\mathbf{h}^{(0)}_{\eta_i} = \pi_{H_0}(\overline{\mathrm{Log}}_\infty \eta_i)$, where π_{H_0} is the projection on H_0 in $\mathbb{R}^n$;
- c is a scaling parameter whose value depends on f_{H_0} (set later to $n^{3/2}/k$).

The condition that the factor base generates Cl_K guarantees that for any challenge ideal there exists a solution to the ClDL on FB. It can be relaxed to some extent to generate only a small index subgroup of Cl_K like in [CDW17]. As we discuss in more details in [BR20, §3.1], the choice of the isometry f_{H_0} is actually not innocuous, and we exhibit in §3.3 a candidate with nice properties.

Finally, we detail in the full version a simpler formalism, viewing L_{phs} as generated by the images of the fundamental elements generating $\mathcal{O}^\times_{K,\mathrm{FB}}$ under the following isomorphism between $\mathcal{O}^\times_{K,\mathrm{FB}}/\mu(\mathcal{O}^\times_K)$ and $L_{\mathsf{phs}} \subsetneq \mathbb{R}^\nu \times \mathbb{Z}^k$:

$$\varphi_{\mathsf{phs}}(\alpha) = \left(c \cdot f_{H_0} \circ \pi_{H_0}(\overline{\mathrm{Log}}_\infty \alpha), \{-v_{\mathfrak{p}_i}(\alpha)\}_{1 \le i \le k}\right). \tag{3.2}$$

Volume of L_{phs} and cardinality of FB. It remains to derive an explicit value for the cardinality k of the factor base FB. As detailed in the full version [BR20]:

$$\operatorname{Vol} L_{\mathsf{phs}} = c^{\nu} \cdot \frac{\sqrt{n}}{2^{r_2/2}} \cdot h_K R_K. \tag{3.3}$$

The idea is then to choose k such that $\operatorname{Vol}^{1/(\nu+k)} = O(1)$, e.g. by taking $(\nu + k) = \ln \operatorname{Vol} L_{\mathsf{phs}}$. Using the analytic class number formula as pointed in §2.4, and using the fact that c will be later set to $n^{3/2}/k$, $\operatorname{Vol} L_{\mathsf{phs}}$ is asymptotically bounded by $\exp \tilde{O}\big(\ln|\Delta_K| + n \ln\ln|\Delta_K|\big)$; therefore, $(\nu + k)$ can be set to:

$$\nu + k = \max\big\{\nu + \log h_K, \ln|\Delta_K| + n \ln\ln|\Delta_K|\big\}. \tag{3.4}$$

The $\log h_K$ part is there as a sufficient but not necessary condition ensuring that Cl_K can be generated by $k \geq \log h_K$ ideals [PHS19a, Lem. 2.7]. As $h_K \leq \tilde{O}(\sqrt{|\Delta_K|})$, we remark that the second term dominates, so the maximum in the above formula can be ignored; in the associated code [PHS19b], $(k + \nu)$ is explicitly set to $\lfloor \ln|\Delta_K| \rfloor$. We stress that in practice the dimension of L_{phs} is quite sensitive to small changes in the value of c or the targeted root volume. We refer to §3.3 for more details and examples.

Preprocessing algorithm. Algorithm 3.1 details the complete preprocessing procedure that, from a number field and some precomputation size parameter, chooses a factor base FB, builds the associated matrix $B_{L_{\mathsf{phs}}}$, and processes L_{phs} in order to facilitate Approx-Cvp queries.

The dimension k of the factor base and the scaling factor c are set in step 1 as in the published code [PHS19b]. Steps 2 and 3 are a concise version of [PHS19a, Alg. 3.1, st. 1–5]; it basically enlarges a generating set of Cl_K of size $k' \leq \log h_K$ by picking $(k - k')$ random prime ideals of bounded norms. The crucial point is to invoke the prime ideal theorem to show that taking a bound which is polynomial in k and $\log|\Delta_K|$ [PHS19a, Cor. 2.10] is actually sufficient.

The last step consists in preprocessing L_{phs} in order to solve Approx-Cvp instances efficiently. As noted in [PHS19a, p.6], the problem is easy without any

Algorithm 3.1 PHS Preprocessing $\mathcal{A}_{\mathsf{pre\text{-}proc}}$

Input: A number field K of degree n and a parameter $\omega \in [0, 1/2]$.
Output: The basis $B_{L_{\mathsf{phs}}}$ with the preimages $\mathcal{O}_{K,\mathrm{FB}}^{\times}$ of its rows, and Laarhoven's hint $\mathcal{V}(L_{\mathsf{phs}})$.

1: Set $k = \big(\lfloor \ln|\Delta_K| \rfloor - \nu\big)$ and $c = \big(n^{3/2}/k\big)$.
2: Compute $\mathrm{Cl}_K = \big\langle [\mathfrak{p}_1], \ldots, [\mathfrak{p}_{k'}] \big\rangle$, with $k' \leq \log h_K$.
3: Randomly extend $\{\mathfrak{p}_1, \ldots, \mathfrak{p}_{k'}\}$ by prime ideals of bounded norm to get $\mathrm{FB} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_k\}$.
4: Compute fundamental elements $\varepsilon_1, \ldots, \varepsilon_\nu, \eta_1, \ldots, \eta_k$ of $\mathcal{O}_{K,\mathrm{FB}}^{\times}$ as in Th. 2.1.
5: Create the matrix $B_{L_{\mathsf{phs}}}$ whose rows are $\varphi_{\mathsf{phs}}(\varepsilon_1), \ldots, \varphi_{\mathsf{phs}}(\eta_k)$ as defined in Eq. (3.1).
6: Use Laarhoven's algorithm to compute a hint $\mathcal{V} = \mathcal{V}(L_{\mathsf{phs}})$ of size $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)}$.
7: **return** $\big(\mathcal{O}_{K,\mathrm{FB}}^{\times}, B_{L_{\mathsf{phs}}}, \mathcal{V}(L_{\mathsf{phs}})\big)$.

constraint on the size of the output hint. To guarantee a hint size that is not exceeding the query phase time, they suggest to use Laarhoven's algorithm [Laa16], which outputs a hint $\mathcal{V}$ of bit-size bounded by $2^{\tilde{O}((\nu+k)^{1-2\omega})}$, i.e. $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)}$ using $(\nu+k)=\tilde{O}(\log|\Delta_K|)$, allowing to deliver the answer for approximation factors $(\nu+k)^{\omega}$ in time bounded by the bit-size of $\mathcal{V}$ [Laa16, Cor. 1–2].

3.2 Query phase: solving id-SVP using the preprocessing

This section describes the query phase $\mathcal{A}_{\mathsf{query}}$ of PHS algorithm; for any challenge ideal $\mathfrak{b}\subseteq K$ having a polynomial description in $\log|\Delta_K|$, it reduces the resolution of Approx-id-SVP in $\mathfrak{b}$ to a single call to an Approx-CVP oracle in L_{phs} as output by the preprocessing phase.

The main idea of this reduction is to multiply the principal ideal output by the ClDL of $\mathfrak{b}$ on FB by ideals in FB until a "better" principal ideal is reached, i.e. having a short generator. In L_{phs}, it translates into adding vectors of L_{phs} to some target vector derived from $\mathfrak{b}$ until the result is short, hence into solving a CVP instance. This is formalized in Alg. 3.2, which rewrites [PHS19a, Alg. 3.2] to take into account our change of conventions in the definition of L_{phs} and the choice of Laarhoven's algorithm as the Approx-CVP oracle [Laa16, §4.2].

Algorithm 3.2 PHS Query $\mathcal{A}_{\mathsf{query}}$

Input: A challenge $\mathfrak{b}$, $\mathcal{A}_{\mathsf{pre\text{-}proc}}(K,\omega)=\left(\mathcal{O}^{\times}_{K,\mathrm{FB}}, B_{L_{\mathsf{phs}}}, \mathcal{V}\right)$, and $\beta>0$ st. for any $\mathbf{t}$, the Approx-CVP oracle using $\mathcal{V}(L_{\mathsf{phs}})$ outputs $\mathbf{w}\in L_{\mathsf{phs}}$ with $\|\mathbf{t}-\mathbf{w}\|_\infty\leq\beta$.
Output: A short element $x\in\mathfrak{b}\setminus\{0\}$.

1: Solve the ClDL for $\mathfrak{b}$ on FB, i.e. find $\alpha\in K$ st. $\langle\alpha\rangle=\mathfrak{b}\cdot\prod_{\mathfrak{p}_i\in\mathrm{FB}}\mathfrak{p}_i^{v_i}$.
2: Define the target as $\mathbf{t}=\left(c\cdot f_{H_0}\circ\pi_{H_0}\left(\overline{\mathrm{Log}}_\infty\,\alpha\right),\left\{-v_i+\beta\right\}_{1\leq i\leq k}\right)$.
3: Use the Approx-CVP solver with $\mathcal{V}(L_{\mathsf{phs}})$ to output $\mathbf{w}\in L_{\mathsf{phs}}$ st. $\|\mathbf{t}-\mathbf{w}\|_\infty\leq\beta$.
4: Compute $s=\varphi_{\mathsf{phs}}^{-1}(\mathbf{w})\in\mathcal{O}^{\times}_{K,\mathrm{FB}}$, using the preimages of $B_{L_{\mathsf{phs}}}$ rows.
5: **return** α/s.

Note that the output of the ClDL in step 1 is a S-unit if and only if $\mathfrak{b}$ is only divisible by prime ideals in the factor base. Each exponent v_i can be expressed as $v_i=v_{\mathfrak{p}_i}(\alpha)-v_{\mathfrak{p}_i}(\mathfrak{b})$. Then, the target defined in step 2 can be viewed as a drifted by β image of α in L_{phs}; using the formalism we introduced in Eq. (3.2), it writes simply as $\mathbf{t}=\varphi_{\mathsf{phs}}(\alpha)+\mathbf{b}_{\mathsf{phs}}$, where $\mathbf{b}_{\mathsf{phs}}=(0,\ldots,0,\beta,\ldots,\beta)$ is non zero only on the k last coordinates. We stress that the role of $\mathbf{b}_{\mathsf{phs}}$ in the definition of the target serves a unique purpose: guarantee that $\alpha/s\in\mathfrak{b}$. In practice, this is not an anecdotic condition, and choosing carefully β has a significant impact on the length of the output, as we will see in §3.3. The rest of the proof of correctness, quality and running time of Alg. 3.2 is recalled in the full version.

3.3 Optimizing PHS parameters

In this section, we propose three improvements of the PHS algorithm. The first one consists in expliciting a candidate for f_{H_0} and using its geometric properties

to derive a smaller lattice dimension, while still guaranteeing the same proven approximation factor. The last two respectively modify the composition of the factor base and the definition of the target vector in a way that drastically improves the approximation factor experimentally achieved by $\mathcal{A}_{\mathsf{query}}$.

Although these improvements do not modify the core of PHS algorithm and have no impact on the asymptotics, they nevertheless are of importance in practice, as we will see in Section 5.

Expliciting the isometry: towards smaller factor bases. We exhibit explicitly a candidate for the isometry f_{H_0} going from $H_0 = \mathbb{R}^n_0 \cap \mathcal{L}_0 \subseteq \mathbb{R}^n$ to $\mathbb{R}^\nu$ and evaluate its effect on the infinity norm. It allows to lower the value of c in Alg. 3.2 from $n\sqrt{n}/k$ to $n(1+\ln n)/k$, inducing a smaller $\operatorname{Vol} L_{\mathsf{phs}}$, and in turn implies using a smaller factor base for the same proven approximation factor. We define the isometry f_{H_0} as the linear map represented by $\overline{\mathrm{GSO}}^{\mathrm{T}}(M_{H_0})$, with:

$$M_{H_0} \overset{\text{def}}{:=} \begin{pmatrix} \nu & \overbrace{\begin{matrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{matrix}}^{\nu+1} \end{pmatrix} \cdot \begin{pmatrix} r_1 & \overbrace{I_{r_1}}^{r_1} & \overbrace{\begin{matrix} \frac{1}{2} & \frac{1}{2} & & \\ & \frac{1}{2} & \frac{1}{2} & \\ & & \ddots & \ddots \\ & & & \frac{1}{2} \quad \frac{1}{2} \end{matrix}}^{2r_2} \\ r_2 & & \end{pmatrix}. \tag{3.5}$$

Actually, M_{H_0} is simply a basis of $\mathbb{R}^n_0 \cap \mathcal{L}_0$ in $\mathbb{R}^n$, constituted of vectors that are orthogonal to $\mathbf{1}_n$ and to each of the r_2 independent vectors $\mathbf{v}_j$, $j \in [\![1, r_2]\!]$, that sends any $\mathbf{y} \in \mathcal{L}_0$ to $\mathbf{0}$ by substracting y_{r_1+2j} from its copy y_{r_1+2j-1} and forgetting every other coordinate.

We prove that this isometry verifies $\forall \mathbf{h} \in H_0$, $\|\mathbf{h}\|_\infty \leq (1+\ln n) \cdot \|f_{H_0}(\mathbf{h})\|_\infty$ [BR20, Pr. 3.2]. Hence, as fully explained in [BR20, §3.3], we can choose:

$$c = \max\left(1, \frac{(1+\ln n)n}{\sum_{\mathfrak{p}\in \mathrm{FB}} \ln \mathcal{N}(\mathfrak{p})}\right). \tag{3.6}$$

We quantify the gain obtained by this new value of c using several experiments, all described and discussed in the full version of this paper [BR20, Tab. 3.1–2].

Lowering the factor base weight. Second, we suggest choosing the k elements of the factor base as the k prime ideals of least possible norm, instead of randomly picking them up to some polynomial bound. As discussed in the full version, this incidentally lowers the approximation factor, which depends on $\prod_{\mathfrak{p}\in \mathrm{FB}} \mathcal{N}(\mathfrak{p})$.

Formally, this only modifies step 3 of Alg. 3.1 as follows. Let $\{\mathfrak{p}_1, \ldots, \mathfrak{p}_{k'}\}$ be a generating set of Cl_K, with $k' \leq \log h_K$, as obtained by the previous step 2. As in Alg. 3.1, using the prime ideal theorem yields that we can choose some bound B polynomial in k and $\log|\Delta_K|$ such that the set of prime ideals of norm bounded by B contains at least k elements. Then, we order this set by increasing

norms, choosing an arbitrary permutation for isonorm ideals, and remove ideals that were already present in $\{\mathfrak{p}_1, \ldots, \mathfrak{p}_{k'}\}$. It remains to extract the first $(k-k')$ elements to obtain our factor base.

There is one issue to consider, namely adapting the justification of [PHS19a, H. 4], relying on L_{phs} being a "somehow random" lattice to derive that $\mu_\infty(L_{\mathsf{phs}})$ is close to $\lambda_1^{(\infty)}(L_{\mathsf{phs}})$. We discuss this in more details for H. 4.8 in §4.2. Moreover, in practice, it is always possible to empirically upper bound the infinity covering radius of L_{phs} to verify that this heuristic holds. For example, as described in [PHS19a, §4.1]: take sufficiently many random samples $\mathbf{t}_i$ in the span of L_{phs} from a continuous Gaussian distribution of sufficiently large deviation; solve Approx-Cvp for the ℓ_2-norm for each of them to obtain vectors $\mathbf{w}_i \in L_{\mathsf{phs}}$ close to $\mathbf{t}_i$; finally, majorate $\mu_\infty(L_{\mathsf{phs}})$ by $\max_i \|\mathbf{t}_i - \mathbf{w}_i\|_\infty$. Then, if the expected heuristic behaviour is too far from this estimate, we could still replace one ideal of FB by an ideal of bigger norm and iterate the process.

Minimizing the target drift. Our last suggested improvement modifies the definition of the target vector to take into account the fact that valuations at prime ideals are integers. Hence, the condition enforcing $\alpha/s \in \mathfrak{b}$, which was written as $\forall \mathfrak{p} \in \mathrm{FB},\ v_\mathfrak{p}(\alpha) - v_\mathfrak{p}(s) \geq 0$, can be replaced by the equivalent requirement that $\forall \mathfrak{p} \in \mathrm{FB},\ v_\mathfrak{p}(\alpha) - v_\mathfrak{p}(s) > -1$. Intuitively, this reduces the valuations at prime ideals of the output element by one on average, hence lowering the approximation factor bound. Formally, using the notations of Alg. 3.2, we only modify the definition of the target $\mathbf{t}$ in step 2 of Alg. 3.2. For any $0 < \varepsilon < 1$, let $\widetilde{\beta} = (\beta - 1 + \varepsilon)$ and let $\widetilde{\mathbf{b}}_{\mathsf{phs}} = (0, \ldots, 0, \widetilde{\beta}, \ldots, \widetilde{\beta})$ with non zero values only on the k last coordinates. The modified target is defined as:

$$\widetilde{\mathbf{t}} = \varphi_{\mathsf{phs}}(\alpha) + \widetilde{\mathbf{b}}_{\mathsf{phs}} = \left(c \cdot f_{H_0} \circ \pi_{H_0}\left(\overline{\mathrm{Log}}_\infty\, \alpha\right), \left\{-v_i + \widetilde{\beta}\right\}_{1 \leq i \leq k}\right). \tag{3.7}$$

The remaining steps of Alg. 3.2 stay unchanged. We have to prove that the output is still correct, i.e. that $\alpha/s \in \mathfrak{b}$, where $\mathbf{w} = \varphi_{\mathsf{phs}}(s) \in L_{\mathsf{phs}}$ verifies $\|\widetilde{\mathbf{t}} - \mathbf{w}\|_\infty \leq \beta$. This is done in the following Pr. 3.2, which adapts [PHS19a, Th. 3.3] to benefit from all the improvements of this section. Its proof is moved to [BR20, Pr. 3.5].

Though this adjustment might seem insignificant at first sight, we stress that the induced gain is of order $\prod_{\mathfrak{p} \in \mathrm{FB}} \mathcal{N}(\mathfrak{p})^{1/n}$, which is roughly subexponential in n, and that its impact is very noticeable experimentally. In fact, the quality of the output is so sensitive to this $\widetilde{\beta}$ that we implemented a dichotomic strategy to find, for each challenge $\mathfrak{b}$, the smallest possible translation $\widetilde{\beta}$ that must be applied to $\varphi_{\mathsf{phs}}(\alpha)$ to ensure $(\alpha/s) \in \mathfrak{b}$.

Proposition 3.2. *Given access to an Approx-*Cvp *oracle that, on any input, output* $\mathbf{w} \in L_{\mathsf{phs}}$ *at infinity distance at most* β*, the modified algorithm* $\mathcal{A}_{\mathsf{query}}$ *using the isometry* f_{H_0} *defined in Eq. (3.5), the value* c *defined in Eq. (3.6), and for any* $0 < \varepsilon < 1$*, the modified target* $\widetilde{\mathbf{t}}$ *defined in Eq. (3.7), computes* $x \in \mathfrak{b} \setminus \{0\}$ *such that:* $\|x\|_2 \leq \sqrt{n} \cdot \mathcal{N}(\mathfrak{b})^{1/n} \cdot \exp\left[\frac{(\beta + \lfloor 2\beta - 1 \rfloor) \cdot \sum_{\mathfrak{p} \in \mathrm{FB}} \ln \mathcal{N}(\mathfrak{p})}{n}\right]$.

4 Twisted-PHS algorithm

Our main contribution is to propose a twisted version of the PHS algorithm. The main modification is to use the natural description of the log-S-unit lattice given in Eq. (2.5) that is deduced from the product formula of Eq. (2.1).

On the theoretical side, we prove that our twisted-PHS algorithm performs at least as well as the original PHS algorithm with the same Cvp solver using a preprocessing hint by Laarhoven. More precisely:

Theorem 4.1. *Let $\omega \in [0, 1/2]$ and K be a number field of degree n and discriminant Δ_K. Assume that a basis of $\mathcal{O}_K$ is known. Under GRH and heuristics H. 4.8 and 4.9, there exist two algorithms $\mathcal{A}^{(Laa)}_{\mathsf{tw\text{-}pcmp}}$ and $\mathcal{A}^{(Laa)}_{\mathsf{tw\text{-}query}}$ such that:*

- *Algorithm $\mathcal{A}^{(Laa)}_{\mathsf{tw\text{-}pcmp}}$ takes as input $\mathcal{O}_K$, runs in time $2^{\tilde{O}(\log|\Delta_K|)}$ and outputs a hint $\mathcal{V}$ of bit-size $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)}$;*
- *Algorithm $\mathcal{A}^{(Laa)}_{\mathsf{tw\text{-}query}}$ takes as inputs any ideal $\mathfrak{b}$ of $\mathcal{O}_K$, whose algebraic norm has bit-size bounded by $2^{\mathrm{poly}(\log|\Delta_K|)}$, and the hint $\mathcal{V}$ output by $\mathcal{A}^{(Laa)}_{\mathsf{tw\text{-}pcmp}}$, runs in time $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)} + \mathrm{T}_{\mathsf{Su}}(K)$, and outputs a non-zero element $x \in \mathfrak{b}$ such that $\|x\|_2 \leq 2^{\tilde{O}(\log^{\omega+1}|\Delta_K|/n)} \cdot \lambda_1(\mathfrak{b})$.*

All the results of this section are fully proven in the full version [BR20, §4].

On the practical side though, experimental evidence given in §5 suggest that we achieve much better approximation factors than expected, and that the given lattice bases are a lot more orthogonal than the ones used in [PHS19a]. Thus, in practice, we propose two alternative algorithms $\mathcal{A}^{(\mathrm{bkz})}_{\mathsf{tw\text{-}pcmp}}$ and $\mathcal{A}^{(\mathrm{np})}_{\mathsf{tw\text{-}query}}$: the former applies a minimal reduction strategy as sole lattice preprocessing, and the latter resorts to Babai's Nearest Plane algorithm for the Cvp solver role.

4.1 Preprocessing of the number field

As for the PHS algorithm, the preprocessing phase consists, from a number field K and a size parameter $\omega \in [0, 1/2]$, in building and preparing a lattice L_{tw} that depends only on the number field and allows to express any Approx-id-Svp instance in K as an Approx-Cvp instance in L_{tw}.

Theoretically, the only difference between the original PHS preprocessing and ours resides in the lattice definition and in the factor base elaboration. Its most significant part still consists in computing a hint of constrained size to facilitate forthcoming Approx-Cvp queries. In practice though, we replace this hint computation by merely a few rounds of BKZ with small block size (see §5). In a quantum setting this removes the only part that is not polynomial in $\ln|\Delta_K|$, and in a classical setting avoids the dominating exponential part.

Defining the lattice L_{tw}: a full-rank version of the log-S-unit lattice. Let $\mathrm{FB} = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_k\}$ be a set of prime ideals generating the class group Cl_K. The lattice L_{tw} used by our twisted-PHS algorithm is basically the log-S-unit

lattice $\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\mathcal{O}_{K,\mathrm{FB}}^{\times}$ wrpt. FB under the flat logarithmic embedding, to which we apply an isometric transformation to obtain a full-rank lattice in $\mathbb{R}^{\nu+k}$.

Formally, L_{tw} is defined as the lattice generated by the images of the fundamental elements generating the S-unit group $\mathcal{O}_{K,\mathrm{FB}}^{\times}$, as given by Th. 2.1, under the following map φ_{tw} from K to $\mathbb{R}^{\nu+k}$:

$$\varphi_{\mathsf{tw}}(\alpha) = f_H \circ \pi_H\big(\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\alpha\big), \tag{4.1}$$

- where f_H is an isometry from $H \subset \mathbb{R}^{n+k}$ to $\mathbb{R}^{\nu+k}$, with H the intersection of the trace zero hyperplane $\mathbb{R}_0^{n+k} = \mathbf{1}_{n+k}^{\perp}$, and of the span of $\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\mathcal{O}_{K,\mathrm{FB}}^{\times}$, i.e. $\mathcal{L} = \big\{\mathbf{y} \in \mathbb{R}^{n+k} : y_{r_1+2i-1} = y_{r_1+2i}, i \in [\![1, r_2]\!]\big\}$;
- π_H is the projection on H, in particular it is the identity on the S-unit group.

This map naturally inherits from the homomorphism properties of $\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}$, i.e. $\varphi_{\mathsf{tw}}(\alpha\alpha') = \varphi_{\mathsf{tw}}(\alpha) + \varphi_{\mathsf{tw}}(\alpha')$ and $\forall \lambda \in \mathbb{Z}$, $\varphi_{\mathsf{tw}}(\alpha^{\lambda}) = \lambda \cdot \varphi_{\mathsf{tw}}(\alpha)$, and also defines an isomorphism between $\mathcal{O}_{K,\mathrm{FB}}^{\times}/\mu\big(\mathcal{O}_K^{\times}\big)$ and L_{tw}.

The isometry f_H must be carefully chosen in order to control its effect on the ℓ_∞-norm. Nevertheless, it should be seen as a technicality allowing to work with tools designed for full-rank lattices. Formally, let f_H be the linear map represented by $\overline{\mathrm{GSO}}^{\mathrm{T}}(M_H)$, which denotes the transpose of the Gram-Schmidt orthonormalization of the following matrix:

$$M_H \overset{\text{def}}{:=} \begin{pmatrix} & \nu+1+k & \\ \nu+k & \begin{matrix} -1 & 1 & & \\ & -1 & 1 & \\ & & \ddots & \ddots \\ & & & -1 \quad 1 \end{matrix} & \end{pmatrix} \cdot \begin{pmatrix} r_1 & 2r_2 & k \\ r_1 & I_{r_1} & \\ r_2 & & \begin{matrix} \frac{1}{2}\ \frac{1}{2} & & \\ & \frac{1}{2}\ \frac{1}{2} & \\ & & \ddots\ \ddots \\ & & \frac{1}{2}\ \frac{1}{2} \end{matrix} \\ k & & I_k \end{pmatrix}. \tag{4.2}$$

Actually, M_H is simply a basis of $\mathbb{R}_0^{n+k} \cap \mathcal{L}$ in $\mathbb{R}^{n+k}$, constituted of vectors that are orthogonal to $\mathbf{1}_{n+k}$ and to each of the r_2 independent vectors $\mathbf{v}_j, j \in [\![1, r_2]\!]$ that sends any $\mathbf{y} \in \mathcal{L}$ to $\mathbf{0}$ by substracting y_{r_1+2j} from its copy y_{r_1+2j-1} and forgetting every other coordinate. Hence, graphically, a row basis of L_{tw} is:

$$B_{L\mathsf{tw}} \overset{\text{def}}{:=} \left[\begin{array}{c|c} \widetilde{\Lambda}_K & 0 \\ \hline \overline{\mathrm{Log}}_{\infty}\,\eta_1 & \\ \vdots & \Big(-v_{\mathfrak{p}_j}(\eta_i)\ln \mathcal{N}(\mathfrak{p}_j)\Big)_{1\le i,j\le k} \\ \overline{\mathrm{Log}}_{\infty}\,\eta_k & \end{array}\right] \cdot \overline{\mathrm{GSO}}^{\mathrm{T}}(M_H), \tag{4.3}$$

where the first part is the basis $\widetilde{\Lambda}_{K,\mathrm{FB}}$ of $\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\mathcal{O}_{K,\mathrm{FB}}^{\times}$ defined in §2.3.

Volume of L_{tw} and optimal factor base choice. First, we evaluate the volume of $L_{\mathsf{tw}} = f_H\big(\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\mathcal{O}_{K,\mathrm{FB}}^{\times}\big)$. As the isometry f_H stabilizes the span of the log-S-unit lattice, it preserves its volume, which is given by Pr. 2.3. Using that ideal classes of FB generate the class group, hence $h_K^{(\mathrm{FB})} = h_K$, yields:

$$\mathrm{Vol}\, L_{\mathsf{tw}} = \sqrt{n+k} \cdot 2^{-r_2/2} \cdot h_K R_K \prod_{1 \le i \le k} \ln \mathcal{N}(\mathfrak{p}_i). \tag{4.4}$$

Certainly, the volume of L_{tw} is growing with the log norms of the factor base prime ideals, but a remarkable property is that this growth is at first slower than the lattice density increase induced by the bigger dimension. The meaning of this is that we can enlarge the factor base to densify our lattice up to an optimal point, after which including new ideals become counter-productive.

Formally, let $V_{k'}$ denote the *reduced* volume $\mathrm{Vol}^{1/(\nu+k')}\, L_{\mathsf{tw}}$ for a factor base of size $k' \ge k_0$, where k_0 is the number of generators of Cl_K. We have:

$$V_{k'+1} = V_{k'} \cdot \left(\sqrt{1+\tfrac{1}{n+k'}} \cdot \frac{\ln \mathcal{N}(\mathfrak{p}_{k'+1})}{V_{k'}}\right)^{1/(\nu+k'+1)}. \tag{4.5}$$

This shows that $V_{k'+1} < V_{k'}$ is equivalent to $\ln \mathcal{N}(\mathfrak{p}_{k'+1}) < V_{k'} \Big/ \sqrt{1+\frac{1}{n+k'}}$. Using this property, Alg. 4.1 outputs a factor base maximizing the density of L_{tw}.

First, for a fixed factor base of size k, we compare the reduced volume V_k of L_{tw} with the reduced volume of L_{phs}, denoted $V_{\mathsf{phs}} \overset{\mathrm{not}}{:=} \left(\sqrt{\frac{n}{2^{r_2}}} \cdot h_K R_K\right)^{1/(\nu+k)}$.

Lemma 4.2. *We have:* $\dfrac{V_k}{V_{\mathsf{phs}}} \le \dfrac{e^{1/ne}}{k} \cdot \sum_{\mathfrak{p} \in \mathrm{FB}} \ln \mathcal{N}(\mathfrak{p})$.

This means that the gap between the reduced volume of the twisted lattice and the reduced volume of the untwisted lattice evolves roughly as the arithmetic mean of the $\ln \mathcal{N}(\mathfrak{p})$. We stress that this bound is valid for *any* k.

Although the reduced volume significantly decreases in the first loop iterations, reaching precisely the minimum value can be very gradual, so that it

Algorithm 4.1 Tw-PHS Factor Base Choice $\mathcal{A}_{\mathsf{tw\text{-}FB}}$

Input: A number field K of degree n.
Output: An optimal factor base FB generating Cl_K that minimizes $\mathrm{Vol}^{1/(\nu+k)}\, L_{\mathsf{tw}}$.

1: Compute $\mathrm{Cl}_K = \langle [\mathfrak{q}_1], \ldots, [\mathfrak{q}_{k_0}] \rangle$, with $k_0 \le \log h_K$.
2: Compute $\mathcal{P}(B) = \{\mathfrak{p}_i : \mathcal{N}(\mathfrak{p}_i) \le B\} \setminus \{\mathfrak{q}_1, \ldots, \mathfrak{q}_{k_0}\}$ ordered by increasing norms, where B is chosen st. $\pi_K(B) = \mathrm{poly}(\ln|\Delta_K|) \ge k_0$.
3: FB $\leftarrow \{\mathfrak{q}_1, \ldots, \mathfrak{q}_{k_0}\}$.
4: $i \leftarrow 0$.
5: **while** $\ln \mathcal{N}(\mathfrak{p}_{i+1}) < V_{k_0+i} \Big/ \sqrt{1+\frac{1}{n+k_0+i}}$ **do**
6: Add $\mathfrak{p}_{i+1}$ to FB.
7: $i \leftarrow i+1$.
8: **end while**
9: **return** FB.

might be clever to early abort the loop in Alg. 4.1 when the gradient is too low, or truncate the output to at most $k' = \tilde{O}(\ln|\Delta_K|)$. We quantify the fact that the density loss is at most constant in the worst case in the following result.

Lemma 4.3. *Let $k' = C\big(\ln|\Delta_K| + n\ln\ln|\Delta_K|\big)$. Let V_{min} be the minimum reduced volume output by $\mathcal{A}_{\text{tw-FB}}$, and suppose V_{min} is reached for some $k > k'$, then $V_{k'} \leq e^{1/C+1/ne} \cdot V_{min}$.*

Proposition 4.4. *Algorithm $\mathcal{A}_{\text{tw-FB}}$ terminates in time $\mathrm{T}_{Su}(K) + \mathrm{poly}(\ln|\Delta_K|)$ and outputs a factor base of size $k = \mathrm{poly}(\ln|\Delta_K|)$ using $B = \mathrm{poly}(\ln|\Delta_K|)$.*

In practice, experiments of §5 report that the dimensions of the factor bases output by $\mathcal{A}_{\text{tw-FB}}$ are significantly smaller than those showed in [BR20, Tab. 3.1–2] for the (optimized) PHS algorithm, so that Lem. 4.3 is never triggered.

Preprocessing algorithm. Algorithm 4.2 details the complete preprocessing procedure that, from a number field and some precomputation size parameter, chooses a factor base FB, builds the associated matrix $B_{L\text{tw}}$, and processes L_{tw} in order to facilitate Approx-Cvp queries.

Algorithm 4.2 Tw-PHS Preprocessing $\mathcal{A}_{\text{tw-pcmp}}$

Input: A number field K of degree n and a parameter $\omega \in [0, 1/2]$ or b.
Output: The basis $B_{L\text{tw}}$ with the preimages $\mathcal{O}_{K,\text{FB}}^{\times}$ of its rows, and Laarhoven's hint $\mathcal{V}(L_{\text{tw}})$.

1: Get an optimal factor base $\text{FB} = \mathcal{A}_{\text{tw-FB}}(K)$ of size $k = \sharp\text{FB}$. If needed, truncate the output to $k = \tilde{O}(\ln|\Delta_K|)$ as in Lem. 4.3.
2: Compute fundamental elements $\varepsilon_1, \ldots, \varepsilon_\nu, \eta_1, \ldots, \eta_k$ of $\mathcal{O}_{K,\text{FB}}^{\times}$ as in Th. 2.1.
3: Create $B_{L\text{tw}}$, whose rows are $\varphi_{\text{tw}}(\varepsilon_1), \ldots, \varphi_{\text{tw}}(\eta_k)$ as defined in Eq. (4.3).
4: Use Laarhoven's algorithm to compute a hint $\mathcal{V} = \mathcal{V}(L_{\text{tw}})$ of size $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)}$.
5: (or) Use a BKZ of small block size to reduce the basis of L_{tw}.
6: **return** $\big(\mathcal{O}_{K,\text{FB}}^{\times}, B_{L\text{tw}}, \mathcal{V}(L_{\text{tw}})\big)$.

This Tw-PHS preprocessing differs from the original PHS preprocessing given in Alg. 3.1 on two aspects: the factor base, output by $\mathcal{A}_{\text{tw-FB}}$ in step 1 and which is essentially much smaller in practice, and the new twisted lattice in step 3.

The last two alternative steps consists in preprocessing L_{tw} in order to solve Approx-Cvp instances efficiently. Theoretically, we retain in step 4 the same approach as in step 6 of the original PHS preprocessing Alg. 3.1, that guarantees a hint size not exceeding the query phase time using Laarhoven's algorithm [Laa16]. This outputs a hint $\mathcal{V}$ of bit size bounded by $2^{\tilde{O}(\nu+k)^{1-2\omega}}$, i.e. $2^{\tilde{O}(\log^{1-2\omega}|\Delta_K|)}$ using $(\nu + k) = \tilde{O}(\log|\Delta_K|)$, allowing to deliver the answer for approximation factors $(\nu + k)^{\omega}$ in time bounded by the bit size of $\mathcal{V}$ [Laa16, Cor. 1–2]. This theoretic version will be denoted by $\mathcal{A}_{\text{tw-pcmp}}^{(\text{Laa})}$.

Nevertheless, in practice the twisted lattice output by Alg. 4.2 incidentally appears to be a lot more orthogonal than expected. That's the reason why we

suggest to replace the exponential step 4 of Alg. 4.2 by step 5, which performs some polynomial lattice reduction using a small block size BKZ. In a quantum setting this removes the only part that is not polynomial in $\ln|\Delta_K|$, and in a classical setting avoids the dominating exponential part. This practical version will be denoted by $\mathcal{A}^{(\mathrm{bkz})}_{\mathsf{tw\text{-}pcmp}}$.

4.2 Query phase

This section describes the query phase $\mathcal{A}_{\mathsf{tw\text{-}query}}$ of the Tw-PHS algorithm. As for the query phase of the original PHS algorithm, it reduces the resolution of Approx-id-SVP in $\mathfrak{b}$, for any challenge ideal $\mathfrak{b} \subseteq K$ having a polynomial description in $\log|\Delta_K|$, to a single call to an Approx-CVP oracle in L_{tw} as output by the preprocessing phase. The main idea of this reduction remains to multiply the principal ideal generator output by the ClDL of $\mathfrak{b}$ on FB by elements of $\mathcal{O}^{\times}_{K,\mathrm{FB}}$ until we reach a principal ideal having a short generator. This translates into adding vectors of L_{tw} to some target vector derived from $\mathfrak{b}$ until the result is short, hence into solving a CVP instance in the log-S-unit lattice L_{tw}.

The essential difference of the Tw-PHS version lies in the definition of this target, which is adapted in order to benefit from the twisted description of the log-S-unit lattice. This is formalized in Alg. 4.3.

Note that the output of the ClDL in step 1 is not a S-unit unless $\mathfrak{b}$ is divisible only by prime ideals of FB. For each i, $v_i = v_{\mathfrak{p}_i}(\alpha) - v_{\mathfrak{p}_i}(\mathfrak{b})$. For convenience and without any loss of generality we shall assume that $\mathfrak{b}$ is coprime with *all* elements of the factor base, i.e. $\forall \mathfrak{p} \in \mathrm{FB}$, $v_{\mathfrak{p}}(\mathfrak{b}) = 0$. In that case, the target in step 2 writes naturally as $\mathbf{t} = \varphi_{\mathsf{tw}}(\alpha) + f_H(\mathbf{b}_{\mathsf{tw}})$. This target definition calls a few comments. First, the output of the ClDL is projected on the whole log-S-unit lattice instead of only on the log-unit sublattice, hence maintaining its length and algebraic norm logarithms in the instance scope. Thus, the way our algorithm uses S-units to reduce the solution of the ClDL problem can be seen as a smooth generalization of the way traditional SGP solvers use regular units to reduce the solution of the PIP as in [CDPR16]. Second, the sole purpose of the drift by $\mathbf{b}_{\mathsf{tw}}$ is to ensure that $\alpha/s \in \mathfrak{b}$. Adapting its definition to the twisted

Algorithm 4.3 Tw-PHS Query $\mathcal{A}_{\mathsf{tw\text{-}query}}$

Input: Challenge $\mathfrak{b}$, $\mathcal{A}_{\mathsf{tw\text{-}pcmp}}(K,\omega) = \left(\mathcal{O}^{\times}_{K,\mathrm{FB}}, B_{L\mathsf{tw}}, \mathcal{V}\right)$, and $\widetilde{\beta} > 0$ st. for any $\mathbf{t}$, the Approx-CVP oracle using $\mathcal{V}(L_{\mathsf{tw}})$ outputs $\mathbf{w} \in L_{\mathsf{tw}}$ with $\|f_H^{-1}(\mathbf{t}-\mathbf{w})\|_\infty \leq \widetilde{\beta}$.
Output: A short element $x \in \mathfrak{b} \setminus \{0\}$.
1: Solve the ClDL for $\mathfrak{b}$ on FB, i.e. find $\alpha \in K$ st. $\langle \alpha \rangle = \mathfrak{b} \cdot \prod_{\mathfrak{p}_i \in \mathrm{FB}} \mathfrak{p}_i^{v_i}$.
2: Define the target $\mathbf{t}$ as $f_H^{-1}(\mathbf{t}) = \pi_H\left(\overline{\mathrm{Log}}_\infty\, \alpha, \left\{-v_i \ln \mathcal{N}(\mathfrak{p}_i)\right\}_{1\leq i\leq k}\right) + \mathbf{b}_{\mathsf{tw}}$, where the drift $\mathbf{b}_{\mathsf{tw}} \in H$ will be defined in Eq. (4.6).
3: Solve Approx-CVP with $\mathcal{V}(L_{\mathsf{tw}})$ to get $\mathbf{w} \in L_{\mathsf{tw}}$ st. $\|f_H^{-1}(\mathbf{t}-\mathbf{w})\|_\infty \leq \widetilde{\beta}$.
4: (or) Use Babai's Nearest Plane to get $\mathbf{w} \in L_{\mathsf{tw}}$ st. $\|f_H^{-1}(\mathbf{t}-\mathbf{w})\|_\infty$ is small.
5: Compute $s = \varphi_{\mathsf{tw}}^{-1}(\mathbf{w}) \in \mathcal{O}^{\times}_{K,\mathrm{FB}}$, using the preimages of the rows of $B_{L\mathsf{tw}}$.
6: **return** α/s.

setting is slightly tedious and deferred to the next paragraph. The most notable novelty is that we force the use of a drift that is *inside* the log-S-unit lattice span. This somehow captures and compensates for the perturbation induced on infinite places for correcting negative valuations on finite places using S-units.

Finally, as already mentioned, L_{tw} seems much more orthogonal *in practice* than expected, so that we advise to resort to Babai's Nearest Plane algorithm for solving Approx-CVP in L_{tw}, instead of using Laarhoven's query phase with the precomputed hint. We only keep Laarhoven's algorithm to theoretically prove the correctness and complexity of our new algorithm. The theoretical and practical versions of $\mathcal{A}_{\mathsf{tw\text{-}query}}$ are respectively denoted by $\mathcal{A}^{(\mathrm{Laa})}_{\mathsf{tw\text{-}query}}$ and $\mathcal{A}^{(\mathrm{np})}_{\mathsf{tw\text{-}query}}$.

We now detail explicitly our target choice, from which we deduce the correctness and the output quality of Alg. 4.3, as fully proven in [BR20].

Definition of the target vector. Recall that we assumed that $\mathfrak{b}$ is coprime with FB, hence $f_H^{-1}(\mathbf{t}) = \pi_H\big(\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\alpha\big) + \mathbf{b}_{\mathsf{tw}}$, for some $\mathbf{b}_{\mathsf{tw}} \in H$ that must ensure $\alpha/s \in \mathfrak{b}$, for $s = \varphi_{\mathsf{tw}}^{-1}(\mathbf{w})$ and when $\|f_H^{-1}(\mathbf{t}-\mathbf{w})\|_\infty \leq \widetilde{\beta}$. Indexing coordinates by places, we exhibit $\mathbf{b}_{\mathsf{tw}} = \big(\{b_\sigma\}_{\sigma\in\mathcal{S}_\infty\cup\overline{\mathcal{S}}_\infty}, \{b_\mathfrak{p}\}_{\mathfrak{p}\in\mathrm{FB}}\big)$, where:

$$\begin{cases} b_\sigma = -\frac{k}{n}\left(\frac{\ln\mathcal{N}(\mathfrak{b})}{n+k} + \widetilde{\beta}\right) + \frac{1}{n}\sum_{\mathfrak{p}\in\mathrm{FB}}\ln\mathcal{N}(\mathfrak{p}) & \text{for } \sigma \in \mathcal{S}_\infty \cup \overline{\mathcal{S}}_\infty, \\ b_\mathfrak{p} = \widetilde{\beta} - \ln\mathcal{N}(\mathfrak{p}) + \frac{\ln\mathcal{N}(\mathfrak{b})}{n+k} & \text{for } \mathfrak{p} \in \mathrm{FB}. \end{cases} \tag{4.6}$$

It is easy to verify that all coordinates sum to 0, i.e. $\mathbf{b}_{\mathsf{tw}} \in H$. We now explain this choice, first showing that under the above hypotheses, Alg. 4.3 is correct.

Proposition 4.5. *Given access to an Approx-*CVP *oracle that on any input* $\mathbf{t}$*, outputs* $\mathbf{w} \in L_{\mathsf{tw}}$ *st.* $\|f_H^{-1}(\mathbf{t}-\mathbf{w})\|_\infty \leq \widetilde{\beta}$*,* $\mathcal{A}_{\mathsf{tw\text{-}query}}$ *outputs* $x \in \mathfrak{b}\setminus\{0\}$*.*

The proof of Pr. 4.5 also quantifies the intuition that the output element has smaller valuations at big norm prime ideals. In particular, strictly positive valuations occur only for ideals st. $\ln\mathcal{N}(\mathfrak{p}) \leq \widetilde{\beta}$. This has a very valuable consequence: estimating the ℓ_∞-norm covering radius of L_{tw} allows to control the prime ideal support of any optimal solution. Hence, even if the Approx-CVP cannot reach $\mu_\infty(L_{\mathsf{tw}})$, it is possible to confine the algebraic norm of each query output by *not* including in FB the prime ideals whose log-norm would *in fine* exceed $\mu_\infty(L_{\mathsf{tw}})$, and at which the optimal solution provably has a null valuation. Roughly speaking, this is what $\mathcal{A}_{\mathsf{tw\text{-}FB}}$ tends to achieve in Alg. 4.1.

Translating infinite coordinates. As already mentionned, one important novelty consists in forcing the drift used to ensure $\alpha/s \in \mathfrak{b}$ to be *inside* the log-S-unit span. The underlying intuition is that "correcting" negative valuations at finite primes should only involve S-units. We modelize this by splitting the weight of the $b_\mathfrak{p}$'s evenly across the infinite places coordinates, hence obtaining Eq. (4.6). This heuristically presumes that S-units absolute value logarithms are generically balanced on infinite places. Let us summarize our target definition:

$$\mathbf{t} = f_H\left(\left\{\alpha_\sigma - \tfrac{1}{n}\left[k\widetilde{\beta} + \ln\mathcal{N}(\mathfrak{b}) - \textstyle\sum_{\mathfrak{p}\in\mathrm{FB}}\ln\mathcal{N}(\mathfrak{p})\right]\right\}_\sigma, \left\{\alpha_\mathfrak{p} + \widetilde{\beta} - \ln\mathcal{N}(\mathfrak{p})\right\}_{\mathfrak{p}\in\mathrm{FB}}\right). \tag{4.7}$$

Quality of the output of $\mathcal{A}^{(\mathbf{Laa})}_{\mathsf{tw\text{-}query}}$. To bound the quality of the output of Alg. 4.3, the general idea is that minimizing the distance of our target to the twisted lattice directly minimizes the p-adic absolute values $-v_{\mathfrak{p}}(\alpha)\ln\mathcal{N}(\mathfrak{p})$ instead of minimizing the valuations $v_{\mathfrak{p}}(\alpha)$ independently of $\ln\mathcal{N}(\mathfrak{p})$.

This makes use of the following log-S-unit lattice structure lemma, adapting its log-unit lattice classical equivalent [PHS19a, Lem. 2.11–12], [CDPR16, §6.1]:

Lemma 4.6. *For $\alpha \in K$, let $\mathbf{h}_\alpha \stackrel{\text{def}}{:=} \pi_H\big(\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\alpha\big)$. Decompose $\langle\alpha\rangle$ on* FB *as $\mathfrak{b}\cdot\prod_{\mathfrak{p}\in\mathrm{FB}}\mathfrak{p}^{v_{\mathfrak{p}}(\alpha)}$, with $\mathfrak{b}$ coprime to* FB*. Then $\overline{\mathrm{Log}}_{\infty,\mathrm{FB}}\,\alpha = \mathbf{h}_\alpha + \frac{\ln\mathcal{N}(\mathfrak{b})}{n+k}\cdot\mathbf{1}_{n+k}$. Furthermore, the length of α is bounded by:*

$$\|\alpha\|_2 \leq \sqrt{n}\cdot\mathcal{N}(\mathfrak{b})^{1/(n+k)}\cdot\exp\Big[\max_{1\leq j\leq n}(\mathbf{h}_\alpha)_j\Big].$$

Note that using the max of the coordinates of $\mathbf{h}_\alpha$ instead of its ℓ_∞-norm norm acknowledges for the fact that logarithms of small infinite valuations can become large negatives that should be ignored when evaluating the length of α.

Theorem 4.7. *Given access to an Approx-*CVP *oracle that on any input $\mathbf{t}$, outputs $\mathbf{w}\in L_{tw}$ st. $\|f_H^{-1}(\mathbf{t}-\mathbf{w})\|_\infty \leq \widetilde{\beta}$, $\mathcal{A}_{tw\text{-}query}$ computes $x\in\mathfrak{b}\setminus\{0\}$ such that $\|x\|_2 \leq \sqrt{n}\cdot\mathcal{N}(\mathfrak{b})^{1/n}\cdot\exp\left[\frac{(n+k)\widetilde{\beta}-\sum_{\mathfrak{p}\in\mathrm{FB}}\ln\mathcal{N}(\mathfrak{p})}{n}\right]$.*

This outperforms the bound of Pr. 3.2 if $(n+k)\cdot\widetilde{\beta} \leq 2\beta\cdot\sum_{\mathfrak{p}\in\mathrm{FB}}\ln\mathcal{N}(\mathfrak{p})$. In particular, this is implied by Lem. 4.2 if $\widetilde{\beta}/\beta \approx V_k/V_{\mathsf{phs}}$ for $k \geq n$. We will see that under some reasonable heuristics, this is indeed the case when using the *same* factor base, and that experiments suggest a much broader gap. One intuitive reason for this behaviour is that the covering radius of our twisted lattice grows at a slower pace than the log-norm of the prime ideals of FB.

Heuristic evaluation of $\widetilde{\beta}$. Proving the second part of Th. 4.1 necessitates to evaluate $\widetilde{\beta}$. This evaluation rely on several heuristics that adapt heuristics [PHS19a, H. 4–6]. We argue that the arguments developped in [PHS19a, §4] to support these heuristics can be transposed to our setting, as fully discussed in the full version, and both heuristics are validated by experiments in §5.

Heuristic 4.8 (Adapted from [PHS19a, H. 4]). The ℓ_∞-norm covering radius of L_{tw} is $O\big(\mathrm{Vol}^{1/(\nu+k)}\,L_{\mathsf{tw}}\big)$. Likewise, $\mu_2(L_{\mathsf{tw}}) = O\big(\sqrt{\nu+k}\cdot\mathrm{Vol}^{1/(\nu+k)}\,L_{\mathsf{tw}}\big)$.

This assumption relies on L_{tw} to behave like a random lattice. Heuristically, prime ideals of FB represent uniform random classes in Cl_K, and S-units archimedean absolute value logarithms are likely to be uniform in $\mathbb{R}^n/\overline{\mathrm{Log}}_\infty\,\mathcal{O}_K^\times$.

Heuristic 4.9 (Adapted from [PHS19a, H. 5–6]). With non-negligible probability over the input target vector $\mathbf{t}$, the vector $\mathbf{w}$ output by Laarhoven's algorithm satisfies $\|f_H^{-1}(\mathbf{t}-\mathbf{w})\|_\infty \leq O\big(\ln(n+k)/\sqrt{n+k}\big)\cdot\|\mathbf{t}-\mathbf{w}\|_2$.

This heuristic conveys the idea that coefficients of the output of Laarhoven's algorithm are somehow balanced, so that $\|\mathbf{w}\|_2 \approx \sqrt{n+k}\cdot\|f_H^{-1}(\mathbf{w})\|_\infty$. In our setting, this is justified by assuming $\mathbf{t}$ is uniformly distributed in $\big(\mathbb{R}\otimes L_{\mathsf{tw}}\big)/L_{\mathsf{tw}}$, and can be randomized by multiplying $\mathfrak{b}$ by small ideals coprime to FB.

5 Experimental data

This is the first time to our knowledge that this type of algorithm is completely implemented and tested for fields of degrees up to 60. As a point of comparison, the experiments of [PHS19a] constructed the log-S-unit lattice L_{phs} for cyclotomic fields of degrees at most 24 and $h_K \leq 3$, all but the last two being principal [PHS19a, Fig. 4.1].

Hardware and library description. All S-units and class group computations, for the log-S-unit lattice description and the ClDL resolution, were performed using Magma v2.24-10 [BCP97].[3] The BKZ reductions and Cvp/Svp computations used fplll v5.3.2 [The16]. All other parts of the experiments rely on SageMath v9.0 [The20]. All the sources and scripts are available as supplementary material on `https://github.com/ob3rnard/Twisted-PHS`. The experiments took less than a week on a server with 36 cores and 768 GB RAM.

Targeted algorithms. We evaluate three algorithms: the original PHS algorithm, as implemented in [PHS19b]; our optimized version Opt-PHS described in §3.3, and our new twisted variant Tw-PHS, which is described in §4. This yields three different lattices, respectively denoted by L_{phs}, L_{opt} and L_{tw}. Note that there are a few differences between [PHS19a] and its implementation in [PHS19b], but we chose to stick to the provided implementation as much as possible.

In order to separate the improvements due to $\mathcal{A}_{\mathsf{tw\text{-}FB}}$ outputting smaller factor bases from those purely induced by our specific use of the product formula to describe the log-S-unit lattice, we also built lattices $L^{(0)}_{\mathsf{phs}}$ and $L^{(0)}_{\mathsf{opt}}$ corresponding to PHS and Opt-PHS algorithms, but using the *same* factor base as L_{tw}.

Number fields. As announced in §2.1, we consider two families of number fields, namely non-principal cyclotomic fields $\mathbb{Q}(\zeta_m)$ of prime conductors $m \in [\![23, 71]\!]$, and NTRU Prime fields $\mathbb{Q}(z_q)$ where z_q is a root of $x^q - x - 1$, for $q \in [\![23, 47]\!]$ prime. These correspond to the range of what is feasible in a reasonable amount of time, as the asymptotics of $\mathrm{T}_{\mathsf{Su}}(K)$ rapidly speak in a classical setting.

For cyclotomic fields, we managed to compute S-units up to $\mathbb{Q}(\zeta_{71})$ for all factor bases in less than a day, and all log-S-unit lattice variants up to $\mathbb{Q}(\zeta_{61})$. For NTRU Prime fields, we managed all computations up to $\mathbb{Q}(z_{47})$.

BKZ reductions and Cvp *solving.* We applied the same reduction strategy to all of our lattices. Namely, lattices of dimension less than 60 were HKZ reduced, while lattices of greater dimension were reduced using at most 300 loops of BKZ with block size 40. This yields reasonably good bases for a small computational cost [CN11, p.2]. Note the loop limit was in practice never hit.

For Cvp computations, we applied with these reduced bases Babai's Nearest Plane algorithm, as described in [Gal12, §18.1, Alg. 26].

[3] Note that SageMath is significantly faster than Magma for computing class groups, but behaves surprisingly poorly when it comes to computing S-units.

Precision issues. Choosing the right bit precision for floating point arithmetic in the experiments is particularly tricky. We generically used at most 500 bits of precision in our experiments (corresponding to the lattice volume logarithm in base 2 plus some extra margin). There are two notable exceptions:

1. The S-units wrpt. FB can have *huge* coefficients. Computing the absolute values of their embeddings must then be performed at very high precision. All our lattice constructions were conducted using 10000 bits of precision.
2. Computing the target involves the challenge and the ClDL solution, whose coefficients are potentially *huge* rational numbers, up 2^{25000} for e.g. $\mathbb{Q}(\zeta_{53})$. As above, we adjust the precision in order to obtain sensible values.

In all cases, once in the log space the resulting high precision data can be rounded back to the generic precision before lattice reduction or CVP computations.

5.1 Geometric characteristics

First, we evaluated the geometric characteristics of each produced lattice, using indicators recalled in §2.5, namely: the root Hermite factor δ_0, the orthogonality defect δ, and the minimum θ_{min} (resp. average θ_{avg}) vector basis angle. Each of these indicators is declined before and after BKZ reduction to compare their evolution. We also evaluated experimentally the relevance of H. 4.8 and 4.9, according to the protocol we detailed in the full version [BR20]. Example results are given in Tab. 5.1 for NTRU Prime fields, aside the lattices dimension $d = \nu + k$ and reduced volume $V^{1/d}$. Extensive data can be found in the full version [BR20, Tab. B.1–2] for both cyclotomic and NTRU primes fields.

Orthogonality indicators. We first remark that the minimum and average vector basis angles seem difficult to interpret. They are slightly better for the NTRU Prime field but it is harder to extract a general tendency for cyclotomic fields.

After a light BKZ reduction, twisted lattices show significantly better root Hermite factor and orthogonality defect than any other log-S-unit lattice representations, *even* when the lattices have the same dimension, i.e. when the same factor base is used. Second, the evolution of the orthogonality defect before and after the reduction is more restricted in the twisted case than in the others. In particular, we observe that the BKZ-reduced versions of $L^{(0)}_{\mathsf{opt}}$ and $L^{(0)}_{\mathsf{phs}}$ have *big-*

		d	$V^{1/d}$	δ_0		δ		θ_{min}		θ_{avg}		μ_2	μ_∞	$\|\cdot\|_\infty/\|\cdot\|_2$	
				–	bkz	–	bkz	–	bkz	–	bkz			real	H. 4.9
$\mathbb{Q}(z_{47})$	L_{tw}	40	4.576	0.913	0.913	1.650	1.358	49	60	82	84	11.04	5.607	0.632	0.519
	$L^{(0)}_{\mathsf{opt}}$	40	6.231	0.938	0.938	4.628	1.915	37	57	81	81	16.59	8.398	0.658	0.583
	$L^{(0)}_{\mathsf{phs}}$	40	12.06	0.951	0.951	7.908	1.946	38	55	81	81	30.85	15.50	0.662	0.583
	L_{opt}	129	1.376	0.981	0.981	6.189	3.632	21	56	80	83	6.575	2.925	0.696	0.427
	L_{phs}	180	1.309	0.989	0.989	10.15	4.527	31	53	80	83	8.022	2.882	0.704	0.387

Table 5.1 – Geometric characteristics of log-S-unit lattices for NTRU Prime field $\mathbb{Q}(z_{47})$.

ger orthogonality defects than the *unreduced* L_{tw}. This last observation is true for all NTRU Prime fields we tested except $\mathbb{Q}(z_{23})$.

These two phenomenons (better values and small variations) are particularly clear for NTRU Prime fields. We remark that in this case, the twisted version of the log-S-unit lattice fully expresses, since for NTRU Prime fields most factor base elements have distinct norms. On the contrary, factor bases for our targeted cyclotomic fields are composed of one (or two, as for $\mathbb{Q}(\zeta_{59})$) Galois orbits whose elements all have the same norm. Finally, we stress that reducing L_{tw} lattices is much faster in practice than reducing $L^{(0)}_{\mathsf{opt}}$ and $L^{(0)}_{\mathsf{phs}}$. This is corroborated by the graphs of the Gram-Schmidt log norms in §5.2.

5.2 Plotting Gram-Schmidt log norms

For our second experiment, we evaluate the Gram-Schmidt norms of each produced lattice. We propose two comparisons, the first one is before and after BKZ reduction to see the evolution of the norms in each case at iso factor bases in Fig. 5.1, and the second one is between the different lattices (after BKZ reduction) in Fig. 5.2. Again, extensive data for other examples can be found in [BR20, §B.2] for both cyclotomic fields and NTRU Prime fields.

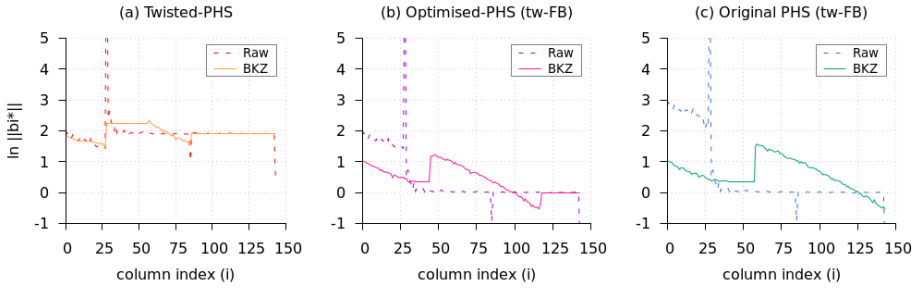

Fig. 5.1 – Log-S-unit lattices for $\mathbb{Q}(\zeta_{59})$: Gram-Schmidt log norms before and after BKZ reduction at iso factor base $\mathcal{A}_{\mathsf{tw\text{-}FB}}(K)$ for: (a) L_{tw}; (b) $L^{(0)}_{\mathsf{opt}}$; (c) $L^{(0)}_{\mathsf{phs}}$.

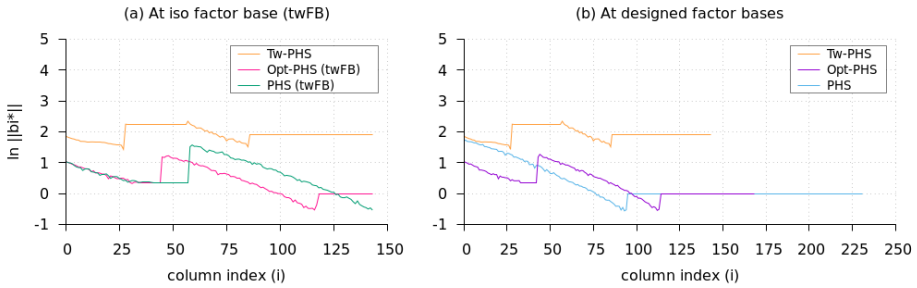

Fig. 5.2 – Log-S-unit lattices for $\mathbb{Q}(\zeta_{59})$: Gram-Schmidt log norms after BKZ reduction: (a) at iso factor base $\mathcal{A}_{\mathsf{tw\text{-}FB}}(K)$; (b) at designed factor bases.

We first remark that in Fig. 5.1 the two curves, before and after BKZ reduction, are almost superposed for the Twisted-PHS lattice. This does not seem to be the case for the two other PHS variants we consider here.

Since the volume of L_{tw} is bigger, by roughly the average log norm of the factor base elements by Lem. 4.2, the Gram-Schmidt log norms of our bases have bigger values. The important phenomenon to consider is how these log norms decrease. Figure 5.2 emphasises that the decrease of the Gram-Schmidt log norms is very limited in the twisted case, compared to other cases (with iso factor base on the left, and the original algorithms on the right), where the decrease of the log norms seems significant. This observation seems to corroborate the fact that the twisted-PHS lattice is already quite orthogonal.

Finally, we note that both phenomenons do not depend on the lattices having the same dimension.

5.3 Approximation factors

We implemented all three algorithms from end to end and used them on numerous challenges to estimate their practically achieved approximation factors. This is to our knowledge the first time that these types of algorithms are completely run on concrete examples.

Ideal Svp *challenges and Cl*dl *computations.* For each targeted field, we chose 50 prime ideals $\mathfrak{b}$ of prime norm q. Indeed, these are the most interesting ideals: in the extreme opposite case, taking $\mathfrak{b}$ inert of norm q^n implies that q reaches the lower bound of Eq. (2.7), as $\|q\|_2 = \sqrt{n} \cdot q$, hence the id-Svp solution is trivial.

We then tried to solve the Cldl for these challenges wrpt. all targeted factor bases. We stress that, using Magma, S-units computations for the Cldl become harder as the norm of the challenge grows. This is especially true when the factor base inflates, hence providing an additional motivation for taking as small as possible factor bases. Therefore, we restricted ourselves to challenges of norms around 100 bits. Computing the Cldl solutions for these challenges revealed much harder than computing S-units on all factor bases, which contain only relatively small prime ideals. As a consequence, we were able to compute the Cldl step only up to $\mathbb{Q}(\zeta_{53})$ (partially) and $\mathbb{Q}(z_{47})$.

Query algorithm. We exclusively used Babai's Nearest Plane algorithm on the BKZ reduced bases of all log-S-unit lattices to solve the Approx-Cvp instances. Actually, the hardest computational task was to compute the output α/s, which necessitates a multi-exponentiation over huge S-units. As a particular point of interest, we stress that using directly the drift proposed in [PHS19a] would be especially unfair. Hence, for a challenge $\mathfrak{b}$, the target drifts $\mathbf{b}_{\mathsf{phs}}$, $\widetilde{\mathbf{b}}_{\mathsf{phs}}$ and $\mathbf{b}_{\mathsf{tw}}$ were all minimized using an iterative dichotomic approach on β and $\widetilde{\beta}$, taking a bigger value if the output $x \notin \mathfrak{b}$, and a smaller value if $x \in \mathfrak{b}$. After 5 iterations, the shortest x that verified $x \in \mathfrak{b}$ is returned.

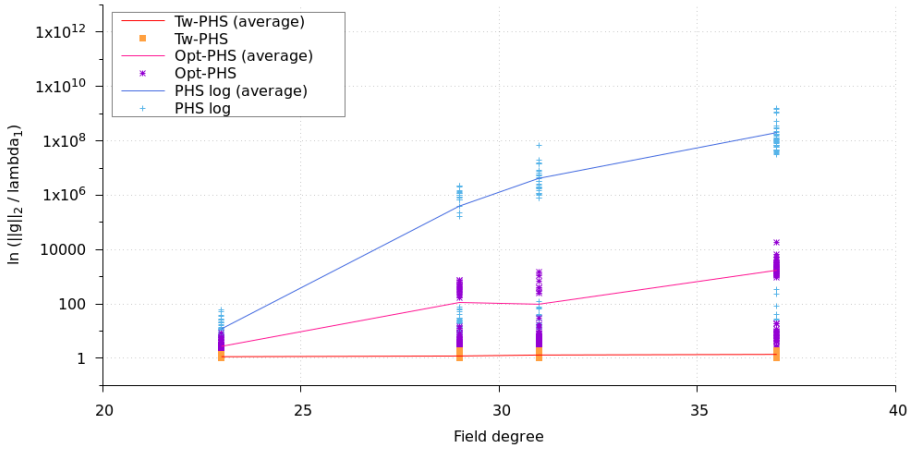

Fig. 5.3 – Approximation factors reached by Tw-PHS, Opt-PHS and PHS for NTRU Prime fields of degrees 23, 29, 31 and 37 (in log scale).

Results. Fig. 1.1 and 5.3 report the obtained approximation factors. Note that for these dimensions, it is still possible to *exactly* solve id-Svp in the Minkowski space, so that these graphs show *real* approximation factors. We stress that we used a logarithmic scale to represent on the same graphs the performances of the Twisted-, Opt-PHS and PHS algorithms. The figures suggest that the approximation factor reached by our algorithm increases very slowly with the dimension, in a way that could reveal subexponential or even better. This feature would be particularly interesting to prove.

As a final remark, we point out that increasing the factor base for our Twisted-PHS algorithm has very little impact on the quality of the output. This is expected, since the log norm of the prime ideals constrain the valuation of the output, as in the proof of Pr. 4.5 [BR20]. On the contrary, increasing the factor base for the PHS and Opt-PHS variants clearly sabotages the quality of their output, as their lattice description is blind to these prime norms.

Acknowledgements. We thank Thomas Ricosset for valuable discussions on the geometry of lattices. Part of this work was performed while the first author was visiting Alice Pellet-Mary and Damien Stehlé at LIP, ENS Lyon for six weeks. This work is supported by the European Union PROMETHEUS project (Horizon 2020 Research and Innovation Program, grant 780701).

References

BCLV17. D. J. Bernstein, C. Chuengsatiansup, T. Lange and C. van Vredendaal: NTRU Prime: Reducing Attack Surface at Low Cost. In *SAC*, vol. 10719 of *LNCS*, pp. 235–260. Springer, 2017.

BCP97. W. Bosma, J. Cannon and C. Playoust: The Magma algebra system. I. The user language. *J. Symbolic Comput.*, **24**(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).

BDPW20. K. d. Boer, L. Ducas, A. Pellet-Mary and B. Wesolowski: Random Self-reducibility of Ideal-SVP via Arakelov Random Walks. Cryptology ePrint Archive, Report 2020/297, 2020.

BEF+17. J. Biasse, T. Espitau, P. Fouque, A. Gélin and P. Kirchner: Computing Generator in Cyclotomic Integer Rings. In *EUROCRYPT (1)*, vol. 10210 of *LNCS*, pp. 60–88. Springer, 2017.

BF14. J. Biasse and C. Fieker: Subexponential class group and unit group computation in large degree number fields. *LMS J. Comp. Math.*, **17**(A):385–403, 2014.

BMT15. D. W. Boyd, G. Martin and M. Thom: Squarefree values of trinomial discriminants. *LMS J. Comput. Math.*, **18**(1):148–169, 2015.

BR20. O. Bernard and A. Roux-Langlois: Twisted-PHS: Using the Product Formula to Solve Approx-SVP in Ideal Lattices (full version). Cryptology ePrint Archive, Report 2020/1081, 2020. https://eprint.iacr.org.

BS16. J.-F. Biasse and F. Song: Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields. In *SODA*, pp. 893–902. SIAM, 2016.

CDPR16. R. Cramer, L. Ducas, C. Peikert and O. Regev: Recovering Short Generators of Principal Ideals in Cyclotomic Rings. In *EUROCRYPT (2)*, vol. 9666 of *LNCS*, pp. 559–585. Springer, 2016.

CDW17. R. Cramer, L. Ducas and B. Wesolowski: Short Stickelberger Class Relations and Application to Ideal-SVP. In *EUROCRYPT (1)*, vol. 10210 of *LNCS*, pp. 324–348. Springer, 2017.

CGS14. P. Campbell, M. Groves and D. Shepherd: Soliloquy: A cau-tionary tale,, 2014. http://docbox.etsi.org/Workshop/2014/201410_CRYPTO/S07_Systems_and_Attacks/S07_Groves_Annex.pdf.

Che13. Y. Chen: *Réduction de réseau et sécurité concrète du chiffrement complètement homomorphe*. Ph.D. thesis, Paris 7, 2013.

CN11. Y. Chen and P. Q. Nguyen: BKZ 2.0: Better Lattice Security Estimates. In *ASIACRYPT*, vol. 7073 of *LNCS*, pp. 1–20. Springer, 2011.

Coh93. H. Cohen: *A course in computational algebraic number theory*, vol. 138 of *Graduate texts in mathematics*. Springer, 1993.

Con. K. Conrad: Ostrowski for number fields. In *Expository papers on Algebraic Number Theory*. https://kconrad.math.uconn.edu/blurbs/gradnumthy/ostrowskinumbfield.pdf.

DPW19. L. Ducas, M. Plançon and B. Wesolowski: On the Shortness of Vectors to Be Found by the Ideal-SVP Quantum Algorithm. In *CRYPTO (1)*, vol. 11692 of *LNCS*, pp. 322–351. Springer, 2019.

EHKS14. K. Eisenträger, S. Hallgren, A. Y. Kitaev and F. Song: A quantum algorithm for computing the unit group of an arbitrary degree number field. In *STOC*, pp. 293–302. ACM, 2014.

Gal12. S. D. Galbraith: *Mathematics of Public Key Cryptography*. Cambridge University Press, 2012.

Gél17. A. Gélin: *Calcul de groupes de classes d'un corps de nombres et applications à la cryptologie*. Ph.D. thesis, UPMC Paris 6, 2017.

GN08. N. Gama and P. Q. Nguyen: Predicting Lattice Reduction. In *EUROCRYPT*, vol. 4965 of *LNCS*, pp. 31–51. Springer, 2008.

HPS98. J. Hoffstein, J. Pipher and J. H. Silverman: NTRU: A Ring-Based Public Key Cryptosystem. In *ANTS*, vol. 1423 of *Lecture Notes in Computer Science*, pp. 267–288. Springer, 1998.

Kom75. K. Komatsu: Integral bases in algebraic number fields. *Journal für die reine und angewandte Mathematik*, **1975**(278-279):137–144, 1975.

Laa16. T. Laarhoven: Sieving for Closest Lattice Vectors (with Preprocessing). In *SAC*, vol. 10532 of *LNCS*, pp. 523–542. Springer, 2016.

LLL82. A. K. Lenstra, H. W. Lenstra and L. Lovász: Factoring Polynomials with Rational Coefficients. *Math. Ann.*, **261**:515–534, 1982.

LPR10. V. Lyubashevsky, C. Peikert and O. Regev: On Ideal Lattices and Learning with Errors over Rings. In *EUROCRYPT*, vol. 6110 of *LNCS*, pp. 1–23. Springer, 2010.

LS15. A. Langlois and D. Stehlé: Worst-case to average-case reductions for module lattices. *Des. Codes Cryptogr.*, **75**(3):565–599, 2015.

MG02. D. Micciancio and S. Goldwasser: *Complexity of Lattice Problems*, vol. 671 of *The Kluwer International Series in Engineering and Computer Science*. Springer, 2002.

Nar04. W. Narkiewicz: *Elementary and Analytic Theory of Algebraic Numbers*. Springer Monographs in Mathematics. Springer, 3rd ed., 2004.

NS06. P. Q. Nguyen and D. Stehlé: LLL on the Average. In *ANTS*, vol. 4076 of *LNCS*, pp. 238–256. Springer, 2006.

NV10. P. Q. Nguyen and B. Vallée, eds.: *The LLL Algorithm*. Information Security and Cryptography. Springer, 2010.

Pei16. C. Peikert: A Decade of Lattice Cryptography. *Foundations and Trends in Theoretical Computer Science*, **10**(4):283–424, 2016.

PHS19a. A. Pellet-Mary, G. Hanrot and D. Stehlé: Approx-SVP in Ideal Lattices with Pre-processing. In *EUROCRYPT (2)*, vol. 11477 of *LNCS*, pp. 685–716. Springer, 2019.

PHS19b. A. Pellet-Mary, G. Hanrot and D. Stehlé: Published code of "Approx-SVP in Ideal Lattices with Pre-processing", 2019. https://apelletm.github.io/code/code-approx-ideal-svp.zip/.

PRS17. C. Peikert, O. Regev and N. Stephens-Davidowitz: Pseudorandomness of ring-LWE for any ring and modulus. In *STOC*, pp. 461–473. ACM, 2017.

Reg05. O. Regev: On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pp. 84–93. ACM, 2005.

Sch87. C. Schnorr: A Hierarchy of Polynomial Time Lattice Basis Reduction Algorithms. *Theor. Comput. Sci.*, **53**:201–224, 1987.

SSTX09. D. Stehlé, R. Steinfeld, K. Tanaka and K. Xagawa: Efficient Public Key Encryption Based on Ideal Lattices. In *ASIACRYPT*, vol. 5912 of *LNCS*, pp. 617–635. Springer, 2009.

Swa62. R. G. Swan: Factorization of polynomials over finite fields. *Pacific J. Math.*, **12**(3):1099–1106, 1962.

The16. The FPLLL development team: fplll, a lattice reduction library, 2016. Available at https://github.com/fplll/fplll.

The20. The Sage Developers: *SageMath, the Sage Mathematics Software System (Version 9.0)*, 2020. Available at https://www.sagemath.org.

Was97. L. C. Washington: *Introduction to Cyclotomic Fields*, vol. 83 of *Graduate Texts in Mathematics*. Springer, 2nd ed., 1997.

Xu13. P. Xu: Experimental quality evaluation of lattice basis reduction methods for decorrelating low-dimensional integer least squares problems. *EURASIP J. Adv. Signal Process.*, **2013**:137–165, 2013.