Quantum Lattice Enumeration in Limited Depth

Nina Bindel[1], Xavier Bonnetain[2], Marcel Tiepelt[3], and Fernando Virdia[4]

[1] SandboxAQ, Palo Alto, CA, USA, nina.bindel@sandboxaq.com
[2] Université de Lorraine, CNRS, Inria, Nancy, France, xavier.bonnetain@inria.fr
[3] KASTEL, Karlsruhe Institute of Technology, Karlsruhe, Germany, marcel.tiepelt@kit.edu
[4] NOVA LINCS, DI, FCT, Univerisdade NOVA de Lisboa, Portugal, f.virdia@gmx.com

Abstract In 2018, Aono *et al.* (ASIACRYPT 2018) proposed to use quantum backtracking algorithms (Montanaro, TOC 2018; Ambainis and Kokainis, STOC 2017) to speedup lattice point enumeration. Quantum lattice sieving algorithms had already been proposed (Laarhoven *et al.*, PQCRYPTO 2013), being shown to provide an asymptotic speedup over classical counterparts, but also to lose competitivity at relevant dimensions to cryptography if practical considerations on quantum computer architecture were taken into account (Albrecht *et al.*, ASIACRYPT 2020). Aono *et al.*'s work argued that quantum walk speedups can be applied to lattice enumeration, achieving at least a quadratic asymptotic speedup *à la* Grover search while not requiring exponential amounts of quantum accessible classical memory, as it is the case for sieving. In this work, we explore how to lower bound the cost of using Aono *et al.*'s techniques on lattice enumeration with extreme cylinder pruning assuming a limit to the maximum depth that a quantum computation can achieve without decohering, with the objective of better understanding the practical applicability of quantum backtracking in lattice cryptanalysis.

Keywords: quantum cryptanalysis · lattice enumeration · post-quantum cryptography · quantum circuits

1 Introduction

Cryptographic constructions based on the hardness of computational problems over algebraic lattices have achieved significant popularity in recent years. Part of the reason for this popularity is the conjectured security of protocols built on them against quantum adversaries, due to the apparent hardness of lattice problems against quantum attacks.

The state-of-the-art attacks on lattice problems usually involve the use of lattice reduction techniques, with block reduction algorithms being the most popular choice [73,74,22,14,54,6,31]. The leading cost of block lattice reduction (and therefore, often, of the attacks overall) comes from solving instances

of the (approximate [50,4]) shortest vector problem (SVP) in high dimension. The leading choice for (approximate) SVP solvers are lattice point enumeration [43,28,32,22,13,3] and lattice point sieving [2,58,48,16,6] algorithms. Due to the central role these algorithms play in the cryptanalysis of lattice-based cryptosystems [51,9,5] and because multiple soon-to-be post-quantum standards are lattice-based [75,52,64], it is crucial to have a clear understanding of their cost.

Enumeration and sieving are originally classical algorithms, with asymptotically different *classical* runtime (namely, $2^{O(n \log n)}$ for enumeration and $2^{O(n)}$ for sieving) and memory cost (namely, $O(\text{poly}(n))$ for enumeration and $2^{O(n)}$ for sieving). Several *quantum* speedups on sieving have been proposed [49,44,20,18]. These algorithms improve upon their classical counterparts using a quantum search or a quantum walk to speed up nearest neighbour subroutines in sieving algorithms. Quantum speedups for enumeration have recevied significantly less attention. Hypothetical quadratic quantum speedups on enumeration were first suggested in [70]. Aono, Nguyen, and Shen [13] demonstrated them by leveraging quantum backtracking techniques [55,10] on the *enumeration tree* constructed internally as part of enumeration.

While applicability of these speedups appears clear in the unbounded-depth logical-qubit model, where quantum computation achieves low error rates for free and does not decohere, our current understanding of quantum computer engineering suggests that this model may be overly optimistic for hypothetical real-world quantum adversaries [63]. For example, Albrecht *et al.* [7] investigate the impact of error correction on quantum lattice sieving, determining that achieving even small quantum speedups over classical sieving in the cryptanalytic regime requires making several optimistic algorithmic and physical assumptions. We are currently not aware of any similar work on the validity of quantum speedups on enumeration in similarly constrained models.

Our contributions. In this paper, we set to investigate whether quantum speedups on lattice enumeration [13] apply to enumeration with extreme cylinder pruning in the *limited quantum depth* setting. In this setting, we assume the availability of error-corrected logical qubits and quantum-accessible classical memory (QRACM) [34,47,40]. However, we also assume a limit MaxDepth to the depth that a quantum circuit can achieve. Computations with higher depth than MaxDepth are assumed to decohere, returning noise.

This setting has been proposed by the National Institute of Standards and Technology (NIST) in their Call for Proposals for post-quantum key encapsulation and digital signatures [60, Sec. 4.A.5]. NIST proposes three different values for MaxDepth, capturing different run-times and quantum computing technology, and propose costs for key-search attacks against block ciphers and collision-search attacks for hash functions in this setting. For our experimental evaluation, we adopt the same MaxDepth limitations, and investigate their effect on quantum enumeration with cylinder pruning, using CRYSTALS-Kyber, the KEM candidate selected for standardisation by NIST, as a case-study.

As our initial results suggest that enumeration trees constructed when attacking Kyber are mostly too large to be directly enumerated quantumly when MaxDepth is considered, we propose a combined classical-quantum enumeration algorithm that allows the use of any available amount of quantum computation, regardless of quantum depth budget limits. We provide a detailed yet generous-to-the-adversary analysis of the runtime costs of this combined attack in terms of quantum depth and number of gates, which we then use to estimate a lower bound on the cost of attacking Kyber.

Our results for combined classical-quantum enumeration suggest that current quantum enumeration with cylinder pruning techniques are unlikely to provide practical speedups against cryptographic instances of lattice problems in a MaxDepth setting. In particular, we identify that unless the distribution of the number of nodes in lattice enumeration trees is subject to significant a Jensen's gap (cf. Def. 1), schemes such as Kyber-768 and Kyber-1024 are likely unaffected by quantum backtracking, even at MaxDepth $= 2^{96}$. The case for Kyber-512 is slightly more nuanced. Indeed, in our model analysis we adopt strict enough lower bounds that quantum speedups appear potentially plausible. However, we emphasize that this does not result in "breaks" or contradictions to Kyber's claims: First, our analysis is excessively generous when underestimating predictable overheads, and second, the Kyber team claims Kyber-512 to be no less secure than AES-128. Black-box quantum speedups are known to apply to the latter for high enough values of MaxDepth, so it would not necessarily be surprising or problematic if analogous speedups were to apply to Kyber-512 too.

In order to produce coherent numbers and not overestimate costs, we attempt to use lower bounds for all quantities. This is not always possible though, and we highlight this in the main body where most appropriate from context, in the form of Simpl. Assms. 1 to 3.

As part of our analysis, we also provide some minor results in the analysis of the structure of non-pruned lattice enumeration trees, which we report in the Supplementary Material, accompanied by matching experiments. We have made our source code used to produce or experimental results, tables, and plots publicly available at `https://github.com/mtiepelt/QuantumLatticeEnumeration`.

Related work. The limited quantum depth budget has been explored in the case of Grover's algorithm against AES and LowMC by Jaques *et al.* [39]. Follow-up work focused on both optimizing the Grover search oracles [79], and improving the search algorithm [24]. Albrecht *et al.* [7] investigated the cost of quantum nearest neighbour search for lattice sieving in the non-free error correction setting. Most significantly, Ambainis and Kokainis [10] expanded on the work of Montanaro [55] by developing a tree size estimation algorithm and applying it to quantum backtracking to obtain an algorithm that performs as a classical algorithm would in the worst case, resulting in meaningul speedups in the case where the expected size of the trees being enumerated is much lower than the upper bound. We do not consider this algorithm as in lattice enumeration we experimentally observe the average tree sizes to closely match the analysis that we use as upper bound.

Open directions. Our work suggests further work in various directions. First, more careful (and hence less generous) designs of quantum circuits used for quantum backtracking. Second, the extension of this work to other pruning techniques, such as discrete pruning [13]. Third, the analysis of the multiplicative Jensen gap (c.f. Def. 1) on lattice enumeration trees.

Roadmap. In § 2 we cover preliminaries. In § 3 we describe our lower bounds on the cost of combined classical-quantum enumeration. In § 4 we estimate the cost of the core circuit used in quantum enumeration. In § 5 we estimate our cost lower bounds applied to the primal lattice attack on Kyber.

2 Preliminaries

We denote the set of integers by $\mathbb{Z}$ and the set of real numbers by $\mathbb{R}$. We denote by log logarithms in base 2. We define $[m] := \{1, ..., m\}$. Furthermore, we write the absolute value of $c \in \mathbb{R}$ as $|c|$, the Euclidean norm of $v \in \mathbb{R}^n$ as $||v|| = (v_1^2 + \cdots + v_n^2)^{1/2}$ and the inner product of v, $w \in \mathbb{R}^n$ as $v \cdot w$. Given a finite set S, we denote by $U(S)$ the uniform distribution over S, and write $s \sim U(S)$ for an element being uniformly distributed in S. To describe asymptotic complexities, we write $\mathcal{O}(\cdot)$ for the big-oh and $\Omega(\cdot)$ for the big-omega notation.

2.1 Lattices

An n-dimensional lattice Λ is a discrete, additive subgroup of $(\mathbb{R}^n, +)$, generated by a set of linearly independent vectors $B = (b_1, \ldots, b_r) \in \mathbb{R}^{n \times r}, b_i \in \mathbb{R}^n$ called a basis. We consider full-rank integer lattices $\Lambda = \{\sum_{i=1}^{r} b_i c_i \mid c_i \in \mathbb{Z}\} \subseteq \mathbb{Z}^n$ with $r = n$ and denote B^* the result of performing Gram-Schmidt orthogonalization on B. We let $\lambda_1(\Lambda) = \min_{v \in \Lambda \setminus \{0\}} ||v||$ denote the first minimum of the lattice. Furthermore, the projection $\pi_{B,i} : \Lambda \to \text{span}(b_1, \cdots, b_{i-1})^{\perp}$ maps a lattice point onto the span orthogonal to the vector space spanned by $b_1, \ldots, b_{i-1}$. We omit the subscript B and write $\pi_i(\cdot)$ when the basis being used is clear from context. Given a lattice Λ of rank n, the set $\pi_i(\Lambda) = \{\pi_i(v) \mid v \in \Lambda\}$ is itself a lattice in $\mathbb{R}^{n-i+1}$. We sometimes refer to such lattices as *projective sublattices*.

Lattice enumeration. Given an input basis B of Λ and an upper bound $\lambda_1(\Lambda) \leq R$ (e.g., $R = ||b_1^*||$), lattice enumeration finds a vector $v \in \Lambda$ such that $||v|| \leq R$. The procedure performs a depth-first-search on a tree consisting of an "empty node" root on level $k = 0$, and the set of projected lattice points of norm at most R in $\pi_{n-k+1}(\Lambda)$ on level $k > 0$. The leaves of the tree on level $k = n$ are lattice points of norm at most R.

As originally proposed, enumeration algorithms iterated over a tree spanning the complete intersection of the lattice with a ball of radius R around the origin [62,43,29,71]. Modern variants restrict the search space by introducing a branch-and-bound methodology, pruning the tree based on a heuristic bound on the norm $||\pi_i(v)||$ of the target's orthogonal projections [32,53,3]. This slightly

reduces the success probability p of the algorithm, since the short vector may be erroneously pruned from the tree, introducing a trade-off between faster traversal speed on the smaller tree versus having to re-run the procedure $\mathcal{O}(p^{-1})$ times on re-randomised versions of the lattice basis. In this work, we focus on the extreme cylinder pruning variant originating from [71, Alg. ENUM] used in [32], with pruning bounds R_k for each level k of the search tree.

Enumeration trees. The key observation behind lattice enumeration algorithms is that orthogonal projections cannot increase the norm of a vector. This means, for a lattice Λ with basis $B = (b_1, \ldots, b_n)$, shortest vector v, and sufficiently large R, that $R \geq ||v|| = ||\pi_1(v)|| \geq ||\pi_2(v)|| \geq \cdots \geq ||\pi_n(v)|| \geq 0$, with $\pi_i(v)$ living in a $(n-i+1)$-dimensional subspace of $\mathbb{R}^n$. Thus, to enumerate vectors in Λ of norm at most R, it is sufficient to enumerate vectors in the lower-rank projections $\pi_i(\Lambda)$ for $i \leq n$, discarding guesses for $\pi_i(v)$ if they are too long. Starting from $i = n$, suppose $\pi_n(v) = g_n$ is guessed correctly for some vector $g_n \in \pi_n(\Lambda)$.[1] The enumeration algorithm then attempts to extend this into a guess $g_{n-1} \in \pi_{n-1}(\Lambda)$ for $\pi_{n-1}(v)$ such that $\pi_n(g_{n-1}) = g_n$. If $||g_{n-1}|| \leq R$,[2] one proceeds similarly trying to extend g_{n-1} into a guess for g_{n-2} for $\pi_{n-2}(v)$ and so on; else one attempts to find a different guess $g'_{n-1} \neq g_{n-1}$ for $\pi_{n-1}(v)$ that is short enough, and if no such vector exists one aborts the search in $\pi_{n-1}(\Lambda)$ and attempts to extend a different guess $g'_n \neq g_n$ for $\pi_n(v)$. Every guess g_i for $\pi_i(v)$ of norm at most R becomes a node in the enumeration tree. A node g_i is the *child* of some guess g_{i+1} for $\pi_{i+1}(v)$ such that $\pi_{i+1}(g_i) = g_{i+1}$. Moreover, every node g_i is the *parent* of guesses $\{g_{i-1} \in \pi_{i-1}(\Lambda) \mid \pi_i(g_{i-1}) = g_i\}$ for $\pi_{i-1}(v)$. Careful computation using the Gram-Schmidt vectors $B^* = (b_1^*, \ldots, b_n^*)$ and coefficients $\mu_{i,j} = b_i \cdot b_j^*/||b_j^*||^2$ for $i > j$, shows that given a lattice vector $v = \sum_{i=1}^n c_i b_i$ where $c_i \in \mathbb{Z}$ for all i, its projections are of the form $\pi_j(v) = \sum_{i=j}^n \alpha_i b_i^*$ where $\alpha_j = c_j + \sum_{i>j} \mu_{i,j}\, c_i$. By orthogonality of the $(b_i^*)_i$, we have $||\pi_j(v)||^2 = |\alpha_j|^2 ||b_j^*||^2 + ||\pi_{j+1}(v)||^2$. Hence, for any guess g_{j+1} for $\pi_{j+1}(v)$, a guess g_j for $\pi_j(v)$ with $\pi_{j+1}(g_j) = g_{j+1}$ must satisfy $|\alpha_j|^2 \leq (R^2 - ||\pi_{j+1}(v)||^2)/||b_j^*||^2$, *i.e.*,

$$\left|c_j + \sum_{i>j} \mu_{i,j}\, c_i\right| \leq \frac{\sqrt{R^2 - ||\pi_{j+1}(v)||^2}}{||b_j^*||} = \frac{\sqrt{R^2 - \sum_{r=j+1}^{n} \left(c_r + \sum_{i=r+1}^{n} \mu_{i,r} c_i\right)^2 ||b_r^*||^2}}{||b_j^*||}. \quad (1)$$

Remark 1. In the case of pruned enumeration, the main difference in the process is that we are given a *pruning set* $\{R_i \mid i \in [n], 0 < R_1 \leq \cdots \leq R_n = R\}$ rather than a single bound R, with R_{n-j+1} replacing R in Eq. (1).

[1] Since lattices are symmetrical around the origin, in practice implementations consider only half of the possible guesses for $\pi_i(v)$.

[2] To unburden notation, we temporarily consider the non-pruned case, where $R_k = R$ for all k.

Expected cost of enumeration. The cost of enumeration is typically estimated to be equal to the number of nodes visited by the algorithm—the "enumeration tree". Let Z_k be the set of nodes on the k^{th} level of the tree (that is, at distance k from the empty root node), H_k be the expected cardinality of Z_k over the distribution of random bases being enumerated, and N the total number of nodes in the tree. The cardinality of Z_k depends on the pruning strategy and on the geometry of the projective sublattices implied by the lattice bases. The expected number of total nodes is $\mathbb{E}[N] = \frac{1}{2}\sum_{k=1}^{n} \mathbb{E}[|Z_k|] = \frac{1}{2}\sum_{k=1}^{n} H_k$, where the expectation is taken over the distribution from which the lattice is being sampled, and the $1/2$ factor is due to exploiting lattices' additive symmetry to avoid unnecessarily visiting half of the tree. Cost estimation for the algorithm then reduces to estimating H_k. This is a standard computation that we perform in detail in App. A, closely following [32,12].

Solving LWE via lattice reduction. Learning with errors (LWE) [66,67] is a popular hardness assumption for constructing post-quantum secure primitives, such as Kyber [75]. One of the main approaches to solving LWE is via block lattice reduction algorithms, such as BKZ [73,74]. After building a *lattice embedding* of dimension n of a given LWE challenge, block reduction algorithms will call $O(\text{poly}(n))$ many times a shortest vector problem (SVP) solver, such as lattice enumeration, in dimension in dimension $\beta < n$. This process results in a better basis for the LWE lattice embedding, that allows the attacker to recover the LWE challenge secret. In our work we consider modern versions of BKZ using early-termination and approximate-SVP solvers, which have been shown [50,4] to be effective while requiring to find a "short enough" lattice vector, rather than the shortest. In order to go from a LWE challenge to an embedding and a choice of BKZ block size β, we use the `lwe-estimator` [8].

2.2 Quantum algorithms

We denote quantum states in the Hilbert space $\mathcal{H}$ with Greek letters $|\varphi\rangle, |\phi\rangle, |\psi\rangle$, and registers as lower case letters $|a\rangle$. We denote by Id the identity operator.

Quantum backtracking. Backtracking search is a depth-first-search algorithm that allows exhaustively finding "marked" leaves on trees. Given a tree $\mathcal{T}$ of height n, we partition the set of nodes into levels $1 \leq k \leq n$, based on their distance from the root node r. Every node $x \in \mathcal{T}$ then corresponds to a partial assignment to a set of variables $x_1, \ldots, x_n$ identifying a path to the node, using $*$ to mark yet unassigned values. For example, the root node r is labelled $(*, \ldots, *)$, nodes on level $k = 1$ are labelled $(*, \ldots, *, \tilde{x}_n)$ for some values of $\tilde{x}_n$, and so forth. The backtracking algorithm needs to be provided an oracle for a predicate function $\mathrm{P} : \mathcal{T} \to \{\textsc{True}, \textsc{False}, \textsc{Indeterminate}\}$ that given a node returns True if it is a marked leaf, False if it can be determined that the node cannot be on the path to a marked leaf, and Indeterminate otherwise. Lattice enumeration algorithms are backtracking algorithms with a predicate $\mathrm{P}_R(v)$ returning False

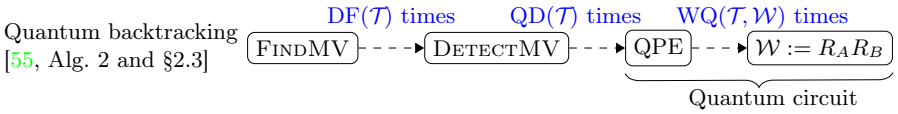


Figure 1: Overview of Montanaro's marked vertex (MV) finding backtracking algorithm [55]. Only part of the algorithm needs to run within MaxDepth.

if $||\pi_i(v)|| > R$, Indeterminate if $||\pi_i(v)|| \leq R$ for $i < n$ and no other value $||\pi_j(v)||$ is known with $j < i$, and True if $||\pi_1(v)|| = ||v|| \leq R$.[3]

In [55], Montanaro introduced quantum backtracking algorithms for detecting [55, Alg. 2] and returning [55, § 2.3] marked vertices in a tree, both achieving an asymptotic speedup over classical backtracking. We depict their main subprocedures in Fig. 1. Both of these algorithms are based on a common quantum walk, QPE($\mathcal{W}$), where $\mathcal{W}$ is the operator that corresponds to a single step of the quantum walk using the predicate P_R to decide if a node is marked.

Then the quantum walk can be seen as a quantum phase estimation (QPE) which consecutively applies WQ($\mathcal{T}, \mathcal{W}$) many times $\mathcal{W}$ to a state corresponding to a superposition of the nodes in the backtracking tree $\mathcal{T}$. By the proof of [55, Lem. 2.4], the eigenvectors of $\mathcal{W}$ are the states that admit a path from the root to a marked node. Measuring the root node after a sufficient number of steps allows to identify whether a path to a marked node exists with false positive probability $\leq 1/4$ and false negative probability $\leq 1/2$.

The detection algorithm DetectMV [55, Alg. 2] consists of repeating $K := \lceil \varepsilon \log(1/\delta_{\mathrm{DMV}}) \rceil$ many times QPE, for some constant $\varepsilon > 0$, to output "marked node exists" or "no marked node exists" with a failure probability of at most δ_{DMV}. We determine bounds on values for ε and δ_{DMV} in § 3.1 (with a more detailed analysis in App. F) and refer to the resulting number of calls to the QPE within DetectMV as QD($\mathcal{T}$).

Theorem 1 ([55, Theorem 1.2], abridged). *Let $\mathcal{T}$ be a backtracking tree of size at most $\#\mathcal{T}^u$ with degree $\mathcal{O}(1)$. Let $\mathrm{P}(x)$ be a predicate that returns* True *if and only if x is a marked node. For any $0 < \delta_{\mathrm{DMV}} < 1$,* DetectMV *outputs "marked node exists" if there exists $x \in \mathcal{T}$ such that $\mathrm{P}(x) =$* True *and "marked node does not exist" otherwise, with failure probability at most δ_{DMV}. The algorithm performs $\mathcal{O}\left(\sqrt{\#\mathcal{T}^u n} \log(1/\delta)\right)$ evaluations of P, using poly(n) qubits.*

For the algorithm returning a marked node in $\mathcal{T}$, FindMV [55, § 2.3], Montanaro suggests to perform classical depth-first-search on $\mathcal{T}$ by using DetectMV as a predicate. DetectMV would be called on the children c_i of the root node, until one on the subtrees rooted at c_i, for some i, returns "marked node exists". It would then proceed to search for a child of c_i spawning a subtree with a marked node, and so on, until reaching a marked leaf. For a tree of height n with nodes having at most d children, FindMV will call $O(nd)$ times DetectMV, in the worst case. We will determine the concrete number of calls (denoted by DF($\mathcal{T}$))

[3] The pruned enumeration case replaces R with R_{n-i+1}, cf. Rmk. 1.

in § 3.1. If no upper bound on the number of nodes of $\mathcal{T}$ is known, the search can be repeated with growing values of $\#\mathcal{T}^u = 2^0, 2^1, 2^2, \ldots$, resulting in an additional runtime factor of $\mathcal{O}(\log \#\mathcal{T})$. Montanaro also shows that overestimating the tree-size does not affect the quantum walk's success probability.

Quantum backtracking for lattice enumeration. Aono *et al.* [13] analyze the asymptotic cost of using quantum backtracking algorithms to perform lattice enumeration in a black-box setting. In [13, Thm. 7(1)], they identify the asymptotic runtime for finding a short non-zero vector using cylinder pruning and $\mathrm{poly}(n)$ many qubits to be $\mathcal{O}\left(\sqrt{\#\mathcal{T}}n^3\ \mathrm{poly}(\log(1/\delta), \log n)\right)$. In [13, Sec 4.2] they argue that this holds also when performing extreme pruning with M randomized lattice bases $(B_i)_{i\leq M}$, by collecting each enumeration tree into a larger, single tree, resulting in an asymptotic runtime for finding a non-zero vector via quantum extreme pruning of $\mathcal{O}\left(\sqrt{\#\mathcal{T}^M}n^3\mathfrak{b}\ \mathrm{poly}(\log n, \log(1/\delta), \log(\mathfrak{b}), \log M)\right)$, where $\mathfrak{b}$ is the bit-size of the entries of B_i, for $i \in [M]$ and $\#\mathcal{T}^M$ is the sum of the number of nodes of all M trees.

2.3 Cost metrics for quantum circuits

Circuit depth. Given a circuit instantiating a quantum algorithm using a given set of gates, a common metric to measure its cost is the circuit's *depth*. This can be defined as the longest path from the input state to the output state, if the circuit is considered as a directed graph with quantum gates as nodes. Circuit depth can be seen as an analogous measure to the runtime of a classical computation, by considering that applying a gate must take a non-zero amount of time, and is therefore often used to express the asymptotic cost of quantum algorithms, as independent quantum gates could be applied in parallel. In this paper, we make two crucial assumptions regarding circuit depth. First, we exclusively measure *T-depth*, that is the circuit depth when only taking into consideration T gates, as preparation of these is expected to be the most time-consuming part of practical quantum computation [42,30]. Second, we investigate the cost of quantum enumeration when imposing a limit MaxDepth to the maximum depth a circuit can achieve while maintaining state coherence. This consideration follows from observing that currently state decoherence seems to be one of the main hurdles to achieving large-scale quantum computation. As part of the call for proposals for its post-quantum cryptography standardization process, NIST [60] proposed the three possible values of 2^{40}, 2^{64}, or 2^{96} for MaxDepth. This limitation means that care should be paid to any circuit parallelization required to stay within MaxDepth circuit depth when measuring the cost of long-running quantum algorithms as these do not always trivially parallelize, such as in the case of Grover's search [78].

Number of gates. Another metric commonly used to express the cost of quantum circuits in the cryptanalysis literature is the number of gates, or *G-cost* (which can always be lower-bounded by the depth of the circuit). The use of the G-cost is motivated by the observation that, in practice, quantum gates are not a physical

device, but an operation performed on the quantum state. Such operation is likely to be managed by a classical microcontroller, meaning that the G-cost is a lower bound on the cost of evaluating gates of a quantum circuit. As such cost also consumes classical resources, it can be compared to the cost of a classical algorithm [41, Def. 2.4].

As part of their call for proposals, NIST [60] defined security *categories* corresponding to the hardness of breaking AES and SHA. When considering quantum algorithms, they expressed this hardness in terms of G-cost, assuming a MaxDepth limit. We will similarly estimate lower bounds on the G-cost of quantum enumeration, in order to simplify comparisons to the rest of the cryptanalytic literature.

Memory. Different kinds of memory devices can be used by quantum computers, categorised depending on how they are interacted with [40]. A common metric that we ignore in this work is that of how many qubits are used by the algorithm. Qubits are currently notoriously to manufacture and maintain in coherent state. Another form of useful memory is QRACM, or *quantum accessible classical memory*. This can be thought of as a classical array $(a_1, \ldots, a_n)$ that can be read by a quantum computer into a state $\sum_{i \leq n} |a_i\rangle$ in $O(n \text{ poly}(\log(n)))$ operations [34,47,40]. All our algorithms use a polynomial amount of qubits, and some of our approaches require an exponential-size QRACM.

Limits of the NIST metrics. The NIST metrics were designed to give a security goal and allow comparisons between candidates. As all metrics, part of them is arbitrary, and they were not designed to accurately define what could be computable in the long term. In particular, the classical and quantum security levels are incomparable, and a break of a system due to a quantum attack does not imply the most efficient way to attack it will be, in the long run, quantum. Still, post-quantum cryptography has to do some optimistic assumptions on the eventual capabilities of quantum computers, as otherwise pre-quantum schemes would be fine.

3 Estimating the Cost of Quantum Enumeration

In this section, we outline the components of our cost estimation of quantum lattice enumeration via backtracking under a MaxDepth restriction. We start by reviewing the gate-cost of Montanaro's FindMV algorithm and the depth of the quantum walk QPE($\mathcal{W}$), since the latter will imply an upper bound to the size of the largest tree that can be searched within a MaxDepth budget for coherent quantum computations. We then proceed to explore the cost of combining quantum enumeration with classical one, to address settings where the trees are too large for the limited quantum depth budget.

3.1 Quantum Backtracking to Find a Marked Vertex

Here, we follow the proof of Thm. 1, aiming to provide concrete lower bounds (rather than asymptotic upper bounds) to the cost of the FindMV algorithm.

The quantum backtracking framework laid out in § 2.2 performs a classical depth-first search on the backtracking tree, where each node is evaluated using multiple, individual quantum walks to decide if its subtree contains a marked node. An overview of Montanaro's quantum algorithm is sketched in Fig. 1. Since the depth of a quantum circuit is the principal limitation for our cost model (see § 2.3) all calls to the quantum circuit QPE($\mathcal{W}$) can be viewed independently, meaning their depth does not accumulate towards the MaxDepth limit.

Node degree. While Thm. 1 assumes the tree being enumerated having constant degree, this is not the case for enumeration trees (of depth n) on general lattices where the leaves are lattice vectors v of norm at most R. Given a node $(\star, \ldots, \star, c_{n-k+1}, \ldots, c_n)$ on level k of the tree, an upper bound $\mathcal{C}_k$ on its degree corresponds to an upper bound on the number of possible values $c_{n-k} \in \mathbb{Z}$ such that $(\star, \ldots, \star, c_{n-k}, c_{n-k+1}, \ldots, c_n)$ is in the backtracking tree. An upper bound on the degree of the tree would then be $\mathcal{C} = \max_k \mathcal{C}_k$. For q-ary lattices as considered in our experiments in § 5, a bound could be $\mathcal{C}_k = q$ for all k. A better bound is given by Lem. 2 in App. B, where we show $\mathcal{C}_k \approx \min(\lfloor 2 \cdot R_{k+1}/||b^*_{n-k}|| \rceil, q)$.

Procedures. As outlined in § 2.2 and Fig. 1, the quantum algorithm that can identify marked vertices in a tree, FindMV, will internally call DetectMV, which detects whether a marked vertex exists in a tree at all. This, in turn, runs quantum phase estimation QPE on the operator $\mathcal{W}$.

Each procedure calls the respective sub-procedure multiple times (with the number of calls depending on the properties of the respective tree $\mathcal{T}$), resulting in total depth and gate cost for FindMV of

$$\text{T-Depth}(\text{FindMV}(\mathcal{T})) = \text{DF}(\mathcal{T}) \cdot \text{QD}(\mathcal{T}) \cdot \text{WQ}(\mathcal{T}, \mathcal{W}) \cdot \text{T-Depth}(\mathcal{W}), \quad (2)$$

$$\text{GCost}(\text{FindMV}(\mathcal{T})) = \text{DF}(\mathcal{T}) \cdot \text{QD}(\mathcal{T}) \cdot \text{WQ}(\mathcal{T}, \mathcal{W}) \cdot \text{GCost}(\mathcal{W}), \quad (3)$$

where DF$(\cdot)$, QD$(\cdot)$, and WQ$(\cdot)$ are the number of calls to the subroutines DetectMV in FindMV, QPE in DetectMV, and $\mathcal{W}$ in QPE, respectively. We will analyze the number of these calls under the assumption that the search tree is of depth n and degree bound by $\mathcal{C}$, prioritising strict lower bounds.

Number of calls DF$(\cdot)$. We define DF$(\mathcal{T})$ to be the number of calls to DetectMV needed within FindMV. Every call to FindMV (cf. § 2.2) performs a classical search upon input tree $\mathcal{T}$, and outputs a single leaf on level n. Aono, Nguyen and Shen [13] analyze the number of calls to DetectMV made when searching an enumeration tree without an asymptotically constant degree. Their analysis performs an asymptotically convenient implicit transformation of the tree into a binary tree [13, Theorem 5], resulting in $O(n \log \mathcal{C})$ calls in the worst case, where we could take $\mathcal{C} = \max_k\{\min(\lfloor 2 \cdot R_{k+1}/||b^*_{n-k}|| \rceil, q)\}$ (cf. App. B). This bound is likely tight on average when trees are guaranteed to contain a marked leaf. However, as we will see in § 3.2, a MaxDepth constraint results

in performing quantum walks on (sub-)trees without such a guarantee. Since FindMV requires a single call to DetectMV to identify that a tree has no marked leafs, we instead lower bound $\mathrm{DF}(\cdot) \geq 1$.

The success probability of a call to FindMV depends on that of DetectMV. In the following paragraph we lower bound the cost of DetectMV with success probability 1, so that FindMV also has full success probability.

Number of calls $QD(\cdot)$. An upper bound on the quantum depth required to run DetectMV can be established directly from [55, Alg. 2] and [55, Lemma 2.4].

Corollary 1 (T-Depth of quantum circuit to detect a marked node). *Let $\mathcal{W}$ be a quantum operator of depth* T-Depth$(\mathcal{W})$ *that acts on a backtracking tree $\mathcal{T}$ in n (unassigned) variables. Let $QPE(\mathcal{W})$ be the quantum circuit performing phase estimation on $\mathcal{W}$. For any failure probability $\delta_{\mathrm{DMV}} \in (0,1)$, there exists a quantum algorithm* DetectMV *that decides with probability at least $1-\delta_{\mathrm{DMV}}$ if a marked node exists in $\mathcal{T}$ by calling QPE $K \coloneqq \lceil \varepsilon \log(1/\delta_{\mathrm{DMV}}) \rceil$ times, such that* T-Depth$(QPE) \leq 1/b\sqrt{\#\mathcal{T} \cdot n} \cdot$ T-Depth$(\mathcal{W})$*, for some value $b > 0$ depending on $\mathcal{W}$.*

As mentioned in § 2.2 and Cor. 1, each call to DetectMV repeats the QPE a total of $K \coloneqq \lceil \varepsilon \log(1/\delta_{\mathrm{DMV}}) \rceil$ times for some constant $\varepsilon > 0$. The value of ε depends on the failure probability of the quantum phase estimation $\mathrm{QPE}(\mathcal{W})$, and on the desired failure probability δ_{DMV} of DetectMV. The failure probability of $\mathrm{QPE}(\mathcal{W})$ in turn depends on the number of applications of the operator $\mathcal{W}$, relative to the tree-size $\#\mathcal{T}$, the dimension n of the lattice and a constant b (cf. Cor. 1). It is important to note that there is a trade-off between the number of repetitions of $\mathcal{W}$ in the QPE, and the number of repetitions of the QPE. We do not consider any optimizations related to this trade-off as they are implementation specific. Instead, since we are determining lower bounds for the number of calls, with $\varepsilon \geq 0$ we lower bound $\mathrm{QD}(\mathcal{T}) \geq 1$.

Number of calls $WQ(\cdot)$. Asymptotically, $\Omega(\sqrt{\#\mathcal{T}n})$ is a lower bound on the query-complexity of detecting a marked node in a tree with $\#\mathcal{T}$ nodes and depth n [1, Theorem 7][56, Sec 4]. Montanaro also notes that Thm. 1.1 and thus Lem. 2.4 of [56] are optimal for $\delta_{\mathrm{DMV}} = \Omega(1)$. As a consequence, Cor. 1 is an asymptotic lower bound if $b \in \Omega(1)$, where in the black box setting T-Depth$(\mathcal{W}) = \Omega(1)$.

Finding the hidden constant for the phase estimation is slightly more involved. While explicit constants exists for phase estimation [17], Montanaro's algorithm may not necessarily use the optimal majority voting scheme as part of DetectMV.

While we investigate a possible approximation to the constants in App. F, for the sake of safety we settle for the following simplifying assumption for our estimations in § 5, believing that this should not cause us to overestimate significantly the runtime of the attack given all the other strict lower bounds we adopt.

Simplifying Assumption 1 *We ignore hidden constants in the query complexity of QPE of and consider* $WQ(\mathcal{T}, \mathcal{W}) = \sqrt{\#\mathcal{T}n}$.

Beyond lower bounds. The lower bounds on $\mathrm{DF}(\mathcal{T})$, $\mathrm{QD}(\mathcal{T})$ and $\mathrm{WQ}(\mathcal{T})$ result a conservative cost estimation, at the cost of underestimating the attack cost. In App. F, we perform a heuristic analysis of the hidden constants, estimating more realistic values for these quantities. Indeed, we show that the number of calls to DetectMV is likely $\mathrm{DF}(\mathcal{T}) \in \{1, n \log \mathcal{C}\}$ depending on the subtree being searched, the a sufficient number of calls to the QPE in DetectMV could be $\mathrm{QD}(\mathcal{T}) = \lceil 20 \log(n \log(\mathcal{C})) \rceil$ for $\varepsilon \leq 20$, and that a sufficient number of calls to $\mathcal{W}$ during QPE is $\mathrm{WQ}(\mathcal{T}) \approx 64\sqrt{\#\mathcal{T}n}$, adopting a constant $b \geq 1/64$.

While our estimations seem realistic, and incidentally support Simpl. Assm. 1, for the sake of keeping our analysis as conservative as possible, we will keep using the strict lower bounds obtained above during attack cost estimation in § 3.3 and 5. Analogue results to those in § 5 using the results in App. F can be found in App. F.1.

Depth of QPE($\mathcal{W}$). With Simpl. Assm. 1 in hand, we can attempt to estimate the depth budget required to break practical lattice-based schemes using quantum enumeration as proposed by Aono, Nguyen and Shen [13]. By using the `lwe-estimator` [8] we obtain the block size β required by the BKZ [73,74] algorithm to successfully run key recovery on Kyber [75] using the primal lattice attack. Using $n = \beta$ and $\mathbb{E}[\#\mathcal{T}]$ equal to the returned cost of enumeration when using a custom cost model implementing the lower bound from [12, Eq. 16], and momentarily assuming $\mathbb{E}[\sqrt{\#\mathcal{T}}] \approx \sqrt{\mathbb{E}[\#\mathcal{T}]}$ (we give a more nuanced analysis in § 3.3), we can see that

$$\log \mathbb{E}[\sqrt{\#\mathcal{T}n}] \approx \frac{\log \mathbb{E}[\#\mathcal{T}] + \log \beta}{2} \approx \begin{cases} 90.3 & \text{for Kyber-512,} \\ 166.2 & \text{for Kyber-768,} \\ 263.7 & \text{for Kyber-1024,} \end{cases}$$

where we assume $\mathcal{T}$ to be a collection of 2^{64} cylinder pruning enumeration trees of dimension β.

While the above numbers are a rule-of-thumb approximation, it can be seen that most likely and regardless of the value of T-Depth($\mathcal{W}$), breaking Kyber-768 and Kyber-1024 with a single direct quantum enumeration will not be possible within MaxDepth $\leq 2^{96}$. To deal with this issue, we propose combining quantum backtracking with classical enumeration, in a manner similar to classical parallel enumeration in the next section.

3.2 Combined Classical-Quantum Enumeration

Generally, parallelization of lattice enumeration [23,38,46] is conceptually simple, as the tree structure directly induces a partitioning to the search problem. This

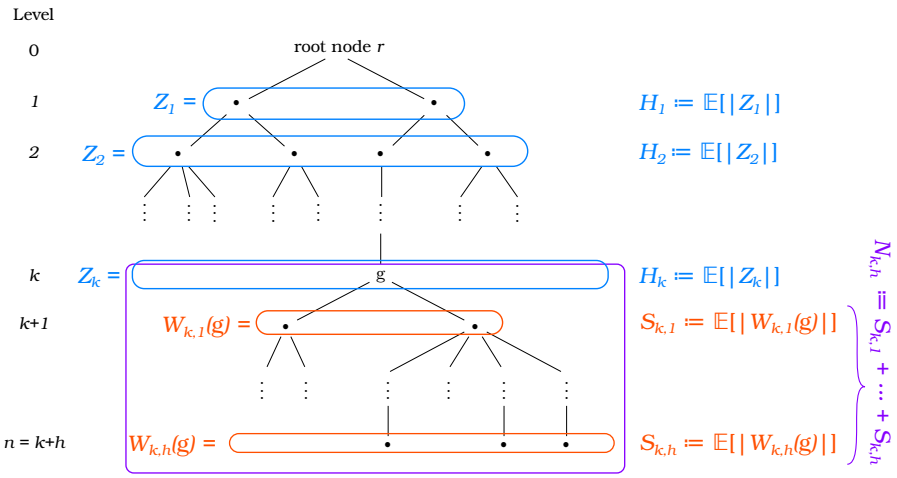


Figure 2: Combined classical-quantum enumeration tree. Quantum enumeration will happen on subtrees rooted at level k, here circled in purple.

means that when searching for short vectors on a tree n levels deep, one can first serially enumerate all nodes on level $k < n$, and then run in parallel lattice enumeration on the subtrees rooted at level k of depth at most $n - k$.

Following this approach, we will run classical enumeration up to depth k, and proceed to run quantum enumeration on the smaller subtrees of depth $h \leq n-k$, corresponding to sub-lattices of dimension h. We will choose k such that the depth of any call to $\mathrm{QPE}(\mathcal{W})$ is within the limit of MaxDepth, following Cor. 1. We depict the general strategy in Fig. 2.

This combined approach is independent of the implementation of the quantum circuits, particularly of the operator $\mathcal{W}$. This means we would be able to estimate bounds on the cost of the attack given different possible values for T-Depth($\mathcal{W}$) and GCost($\mathcal{W}$), including generous lower bounds (cf. § 4.1).

The approach is also compatible with pruned enumeration techniques. In the remainder of the paper, we will focus in particular on cylinder pruning [32]. We will start by analyzing the cost of the combined enumeration algorithm on a single (possibly pruned) tree, in § 3.3, and extend this to the case where M pruned trees are combined to achieve high success probability, in § 3.4.

3.3 Combined Enumeration of a Single Tree

We start by recalling some notation introduced in § 2.1 to describe enumeration trees, illustrated in Fig. 2. Given a tree of depth n, its nodes are partitioned into sets $(Z_i)_{i=1}^n$ of expected cardinality $(H_i)_{i=1}^n$ over the distribution of lattices being inspected, where Z_i collects all the nodes on "level i", that is distant i from the root node r. Any node $g \in Z_k$ generates a subtree $\mathcal{T}(g)$ of depth $h \leq n - k$. The nodes of this subtree are partitioned into sets $(W_{k,i}(g))_{i=1}^h$ of

expected cardinality $(S_{k,i})_{i=1}^{h}$ over random trees and g distributed uniformly in Z_k. The expected number of nodes in the subtree $\mathcal{T}(g)$, including the root g, is $1 + N_{k,h}$, where $N_{k,h} \coloneqq \sum_{i=1}^{h} S_{k,i}$, while the expected number of nodes in the entire enumeration tree $\mathcal{T}$, including the root r, equals $1 + \frac{1}{2}\sum_{i=1}^{n} H_i$, where the $\frac{1}{2}$ factor comes from the fact that the tree is symmetric around 0, and hence only half of the tree needs to be searched to identify all the vectors within the enumeration radius, up to sign.

Classically and quantumly enumerated components. We first discuss how we divide the classical and quantum components of the enumeration algorithm.

Classical component. In the setting of combined classical-quantum enumeration, let k be the level up to where classical enumeration is performed. This costs

$$\mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}[\text{Classical GCost}] \approx 1 + \frac{1}{2}\sum_{i=1}^{k} H_i, \tag{4}$$

where we equate the cost of enumeration to the number of nodes visited by the algorithm, as is standard in the literature.

Quantum component. After the classical enumeration phase, we have H_k nodes on level k, each admitting a subtree of height $h \leq n - k$, and covering the remaining levels of the full enumeration tree. A natural approach could be to enumerate all H_k subtrees individually (and possibly in parallel). However, it is not known which subtree contains the (likely few) marked nodes, meaning that we would be running $\approx H_k$ calls to quantum enumeration. In both pruned and non-pruned enumeration, the bulk of the nodes contained in the trees being traversed is contained in the "middle" levels Z_i for $i \approx n/2$, with pruning "spreading the bulk" on a larger window of levels around $n/2$ [32, Fig. 1]. For our setting, this would imply three scenarios, depending on k:

- $k \approx 1$**,** in this case most of the tree fits within the quantum enumeration subroutine, and a quadratic speedup is possibly achievable,
- $k \approx n/2$**,** in which case we would be running $\approx H_{n/2}$ quantum enumeration calls, meaning that the GCost of the operation would be proportional to $H_{n/2}$, which is approximately the cost of fully-classical enumeration.
- $k \approx n$**,** which means that we would be running some quantum enumeration, but the bulk of the enumeration would anyway be classical, nullifying any possible speedups from quantum enumeration.

While the case $k \approx 1$ is possible, requiring to have a high enough MaxDepth to fit most of the enumeration tree within the quantum subroutine available, is quite restricting; in particular, given the rule-of-thumb numbers from § 3.1.

A possible alternative approach for the quantum phase of the attack is to collect all the subtrees rooted on level k under a single tree, by adding a "virtual"

root node as the "parent" to all the nodes in Z_k, and to run quantum enumeration on this tree; we illustrate this in Fig. 3 for multiple virtual nodes. This approach has the advantage of running a single quantum enumeration rather than H_k many, potentially achieving a better speedup than in the previous case. However, this comes at a cost in terms of *quantum-readable classical memory* (QRACM) [34,47,40], since the $g_i \in Z_k$ would need to be first enumerated and then stored in memory, to be accessible for the quantum algorithm. Except for $k \approx 1$ or $k \approx n$, this approach may require a super-exponential amount H_k of QRACM for any meaningfully small value of k such that a speedup can be achieved (say, $k \lessapprox n/2$).

Given the issues of the two methods above, we consider an interpolation of both. We assume to have access to enough QRACM to store 2^y nodes on level k at a time. We then classically enumerate up to 2^y nodes $g_i \in Z_k$, and collect them under a virtual root node g_r as to form a tree $\mathcal{T}(g_r)$. We then run quantum enumeration on $\mathcal{T}(g_r)$. This means that we will encounter a quantum GCost of

$$\begin{aligned}\mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}[\text{Quantum GCost}] &\approx \mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}\Big[\sum_{\substack{g_r,\text{ out of the}\\ H_k/2^y \text{ many}}} \text{GCost}(\text{FindMV}(\mathcal{T}(g_r)))\Big] \\ &= \frac{H_k}{2^y} \cdot \mathbb{E}[\text{GCost}(\text{FindMV}(\mathcal{T}(g_r)))] \quad \text{(by Wald's identity).} \end{aligned} \tag{5}$$

Simplifying Assumption 2 *The expectation in Eq. (5) is taken over the distribution of random trees $\mathcal{T}$, assuming the $\mathcal{T}(g_r)$ are formed by uniformly sampling 2^y nodes $g_i \in Z_k$ and collecting these under the virtual node g_r.*

Remark 2. Crucially, we note that if the enumeration bound R is small enough to guarantee few marked leaves in the full enumeration tree, then these subtrees would be likely to contain no marked leaf, and hence to be of height strictly smaller than $n-k$. It also means that DetectMV would be called only once at the root node for most calls to $\text{FindMV}(\mathcal{T}(g_r))$ for most g_r (cf. § 3.1).

Expected cost of one quantum enumeration. After having illustrated how to divide the classical and quantum components, we move our attention to computing the expected gate-cost of FindMV. Here, we repurpose the analysis from § 3.1, adapting it to the case where the enumerated tree is rooted on level k and has depth $h \leq n-k+1$.[4]

We start by recalling Eq. (3) for the gate-cost of FindMV for a tree $\mathcal{T}(g)$ of height h and with at most $\mathcal{C}$ children per node:

$$\begin{aligned}\text{GCost}(\text{FindMV}(\mathcal{T}(g))) &= \text{DF}(\mathcal{T}(g)) \cdot \text{QD}(\mathcal{T}(g)) \cdot \text{WQ}(\mathcal{T}(g), \mathcal{W}) \cdot \text{GCost}(\mathcal{W}) \\ &\geq \sqrt{\#\mathcal{T}(g) \cdot h} \cdot \text{GCost}(\mathcal{W}).\end{aligned}$$

[4] The $+1$ term coming from adding a "virtual" root node.

Remark 3. We note that although most subtrees $\mathcal{T}(g)$ may not have height $n-k+1$ due to likely not having any marked leaves (cf. Rmk. 2). Nonetheless we need to assume "full height" $n-k+1$, since the few trees containing marked leaves are of this height. Underestimating the tree height means that the marked leaves would not be found. Hence, every phase estimation of $\mathcal{W}$ assumes a subtree of full height. On the other hand, as we are aiming for strict lower bounds, we do not consider the full height of the implicit binary tree from Aono *et al* [13].

The GCost of operator $\mathcal{W}$ and the expected number of repetitions are independent of each other, and thus, the respective cost can be analyzed individually. The cost of the former is explored in § 4, while we elaborate on the number of repetitions for a single tree next.

To compute $\mathbb{E}\left[\text{GCost}(\text{FindMV}(\mathcal{T}(g)))\right]$ we first notice that $\text{DF}(\mathcal{T}(g))$ and $\text{QD}(\mathcal{T}(g))$ are constant quantities in our analysis (namely, we set both to 1), although in general they likely depend on the lattice problem and our setup of the full algorithm (such as in the choice of k, which would be done a priori based on cost estimations). Similarly we set $\sqrt{h} = \sqrt{n-k+1}$ (cf. Rmk. 3). Hence, $\text{DF}(\mathcal{T}(g))$, $\text{QD}(\mathcal{T}(g))$, and h do not have a probability distribution, and do not affect the computation of the expectation. Similarly, the design of $\mathcal{W}$ is done *a priori*, and thus, the resulting $\text{GCost}(\mathcal{W})$ is a constant (cf. § 4).

This leaves us with having to estimate $\mathbb{E}[\sqrt{\#\mathcal{T}(g)}]$. As pointed out in [13], the probability distribution of the number of nodes in enumeration trees (or subtrees, such as $\mathcal{T}(g)$) is not known. Jensen's inequality gives an upper bound $\sqrt{\mathbb{E}[\#\mathcal{T}(g)]}$ but no clear lower bound. We address this by defining the *multiplicative Jensen's gap*, and evaluate the cost of the attack for different values of it.

Definition 1 (Multiplicative Jensen's gap). *Let X be a random variable. We say X has multiplicative Jensen's gap 2^z if $\sqrt{\mathbb{E}[X]} = 2^z\,\mathbb{E}[\sqrt{X}]$.*

This leaves us to having to estimate $\mathbb{E}[\#\mathcal{T}(g)]$. Given a "virtual" node g collecting 2^y subtrees rooted at nodes $\{g_1, \ldots, g_{2^y}\} \subset Z_k$, by linearity of expectations

$$\mathop{\mathbb{E}}_{\mathcal{T},\{g_i\}_i}[\#\mathcal{T}(g)] = 1 + \sum_{i=1}^{2^y} \mathop{\mathbb{E}}_{\mathcal{T},\{g_i\}_i}[\#\mathcal{T}(g_i)] = 1 + 2^y\,(1 + N_{k,h}) = 1 + 2^y + 2^y \sum_{j=1}^{h} S_{k,j},$$

where the expectation is taken over the distribution of random trees $\mathcal{T}$, and $\{g_1, \ldots, g_{2^y}\}$ is assumed to be a set of uniformly random elements of Z_k. While this may not be exactly true, as during enumeration it may be easier to find "related" elements in Z_k, where their coefficient vectors are similar, we hope this gives a good approximation of the cost.

To lower bound $S_{k,h}$, we use the following result.

Lemma 1. *Given a random enumeration tree generated as part of BKZ-β reduction for β large enough such that the Gaussian heuristic applies,*[5] *a level*

[5] Experimentally, $\beta > 45$ appears to be sufficient.

$k \geq 1$, and a node $g \in Z_k$, then the expected number of nodes in level $k+i$ descending from g is $\mathop{\mathbb{E}}_{\mathcal{T},g}[|W_{k,h}(g)|] = S_{k,h} \geq \frac{H_{k+h}}{H_k}$, where the expectation is taken over the distribution of random trees $\mathcal{T}$, and g is uniformly distributed in Z_k.

Proof. See App. B.1. □

Simplifying Assumption 3 *As part of our analysis, we use Lem. 1 to lower-bound $S_{k,h}$ by the ratio H_{k+h}/H_k, where H_k is estimated as in App. A. At the moment of computing these numbers to estimate the cost of the attack, we will make use of lower bounds on H_k obtained in [12], since closed forms for the exact expectation H_k are notoriously difficult to obtain. This poses a problem at the moment of estimating a lower bound to H_{k+h}/H_k, since this would require the use an upper bound for the denominator. Unfortunately, upper bounds to the size of cylinder pruning enumeration trees appear not to be easy to obtain [12], and therefore as a simplifying measure we use the use the same lower-bound estimate used for the numerator. We remark that the lower bounds we use are measured to be relatively tight [12], suggesting that this should not introduce significant error.*

Combining all the steps above gives us a final estimation of

$$\begin{aligned}\mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}[\text{Quantum GCost}] &\approx \frac{H_k}{2^y} \cdot \mathbb{E}\left[\text{GCost}(\text{FindMV}(\mathcal{T}(g)))\right] \\ &\geq \frac{H_k}{2^y}\left(\frac{1}{2^z}\sqrt{\left(1+2^y+2^y\sum_{j=1}^{n-k+1}\frac{H_{k+j}}{H_k}\right)(n-k+1)}\right)\cdot \text{GCost}(\mathcal{W}), \quad (6)\end{aligned}$$

reducing the estimation of a lower bound on the expected cost to the estimation of H_i and $\text{GCost}(\mathcal{W})$. The estimation of H_i can be done using standard lattice cryptanalysis techniques; we provide a detailed derivation in the case of no pruning and of extreme cylinder pruning in App. A. Estimations for $\text{GCost}(\mathcal{W})$ are discussed in § 4.1.

The same approach can be used to determine the depth of quantum phase estimation of $\mathcal{W}$, which is the limiting factor in a runtime analysis with limited depth budget, resulting in

$$\text{T-Depth}(\text{QPE}(\mathcal{W})) \geq \frac{1}{2^z}\sqrt{\left(1+2^y+2^y\sum_{j=1}^{n-k+1}\frac{H_{k+j}}{H_k}\right)(n-k+1)}\cdot\text{T-Depth}(\mathcal{W}). \quad (7)$$

3.4 Combined Enumeration of a Set of Trees

In the context of enumeration with extreme pruning [32] one considers a trade-off between the success probability of finding a short vector and the number of nodes pruned. Let p be the probability that the enumeration tree contains a node corresponding to a sufficiently short lattice point, that is $p = 1$ if the full tree is

enumerated and $p < 1$ if branches are pruned. In the case of extreme pruning, $p \ll 1$, meaning that enumeration of the tree is much cheaper, but likely to fail. To boost the success probability, the original lattice basis is rerandomized M times, for some large value of M. Under assumptions of independence between the resulting rerandomized, pruned, trees, the probability of finding a short vector in at least one of the M pruned trees is $1-(1-p)^M = 1-(1-Mp+\mathcal{O}(p^2))$, which is high if $M \approx 1/p$ as $M \gg 1$. It should be noticed that in practice rerandomization usually lowers the quality of the bases, essentially "undoing" some of the lattice reduction [3, §2.5]. However, we will ignore this effect for the sake of finding a simple lower bound and assume that the quality of rerandomized bases for the same lattice is the same as the one for the original basis. Moreover, we assume that pruned trees corresponding to these rerandomized bases are independent of each other.

For our purposes, we will collect the M trees corresponding to M bases into a single tree $\mathcal{T}^M$, by adding a top "super-tree" connecting their roots to an overall root. Let r be the root node of this new tree and let $r_1, r_2, ..., r_M$ be the root nodes of the enumeration trees with randomized basis $(B_i)_{i=1}^M$. We arrange r as parent node of the r_i; a sketch of the full tree is illustrated in Fig. 3. The backtracking predicate (cf. § 2.2) that decides on branching on input of a node r_i will always returns INDETERMINATE on the levels 0 and 1, since all enumeration subtrees rooted at the respective r_i are independent of each other due to basis re-randomisation. We define a quantity H_k^M counting the expected number of nodes on level k of $\mathcal{T}^M$, in terms of $H_k^{(i)}$, that is $\mathbb{E}[\#\{\text{nodes on level } k \text{ of } \mathcal{T}(r_i)\}]$ (or "H_k from tree $\mathcal{T}(r_i)$"), where $\mathcal{T}(r_i)$ is the pruned enumeration tree rooted at r_i. If follows that $H_0^M = 1$, $H_1^M = M$, and $H_k^M = \sum_{i\in[M]} H_{k-1}^{(i)} = M \cdot H_{k-1}$ if $k > 1$. From H_k^M we can then redefine $S_{k,h}^M$ similarly to $S_{k,h}$, reducing it to H_k. This means that we can "port" our cost formulae from § 3.3 by replacing H_k with H_k^M. We estimate H_k^M for no pruning and of extreme cylinder pruning in App. A, since this is a standard computation taken from [32,12], and continue with the cost estimation GCOST($\mathcal{W}$) in the next subsection.

4 Instantiations for the quantum operator $\mathcal{W}$

In this section, we focus on exploring lower and upper bound costs for the quantum operator $\mathcal{W}$. At its core, quantum enumeration consists of multiple repetitions of QPE on the operator $\mathcal{W}$ (see Fig. 1). In estimating whether quantum enumeration could be leveraged under a MAXDEPTH constraint, $\mathcal{W}$ plays two roles. First, its depth T-DEPTH($\mathcal{W}$) plays a part in determining how much classical *versus* quantum enumeration is used, by constraining the admissible values for k, y and z (cf. Eq. (7)) based on the requirement that T-DEPTH(QPE($\mathcal{W}$)) $\leq$ MAXDEPTH. Second, its gate-cost GCOST($\mathcal{W}$) is a factor in estimating the total cost of the attack. By Cor. 1, the gate depth of QPE is partly determined by the gate depth of the operator $\mathcal{W}$.

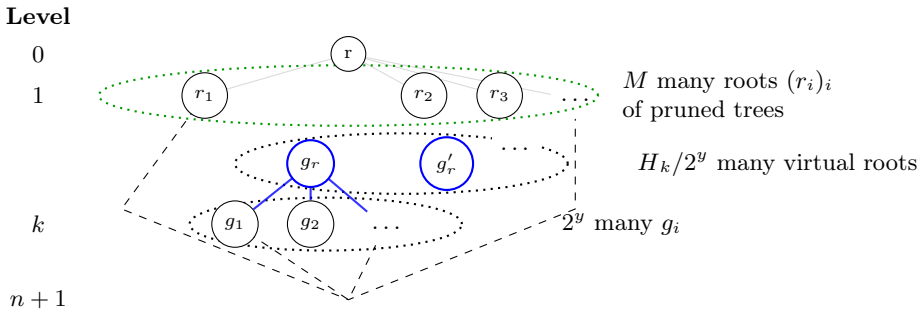


Figure 3: Full enumeration tree $\mathcal{T}^M$ for combined classical-quantum enumeration. Nodes and links in blue correspond to "virtual" nodes collecting 2^y subtrees, as in § 3.3.

4.1 Query-based

The query-based model is assuming access to a black-box oracle that computes the operator $\mathcal{W}$ on any input on-demand. Therefore, we account any query to the operator as having "unit cost", meaning $\text{T-DEPTH}(\mathcal{W}) = \text{GCOST}(\mathcal{W}) = 1$. This setting implies a conservative lower bound on the cost of quantum enumeration. It also represents the setting where classical-quantum enumeration can make the most of any hypothetical quantum speedups.

4.2 Circuit-based: minimal circuit

Here, we aim to estimate the size of a *minimal* or *smallest known* quantum operator $\mathcal{W}$ in the circuit-model. The objective is to provide a more realistic lowest-known bound on the gate cost (*i.e.*, number of T gates) of $\mathcal{W}$ than the very conservative query-based model. Our focus is on finding an approximate smallest T count and depth for arithmetic operations used inside of $\mathcal{W}$. We claim that, at a minimum, the operator $\mathcal{W}$ has to implement a predicate P deciding whether a projected lattice point has length larger than the pruning bound R_i. We assume that each coefficient of the state vector $|c_i\rangle$ consists of $\mathfrak{b} = \lceil \log q \rceil$ qubits. Whenever we require floating point arithmetic, we follow [38] assuming double precision is used, meaning $\xi = 53$ bits of precision are required. To estimate a lower bound to the cost of the minimal circuit for $\mathcal{W}$, we ignore the cost for all operations except for the bare minimum arithmetic that is required to compute the single, most expensive lattice point projection.

Size of quantum arithmetic circuits. The quantum arithmetic literature contains many design proposals for integer and floating point adders and multipliers. We report the smallest to our knowledge in Tab. 1. During our computations, we will be ignoring constants and lower order terms hidden by the $\mathcal{O}$. We refer the reader to App. D for a more detailed literature review.

Table 1: Used T-gate count and depth asymptotics for quantum arithmetic circuits on x-bit numbers.

Operation	T-Depth	GCost	Reference
Addition and Comparison	$\mathcal{O}(\log x)$	$\mathcal{O}(x)$	[26] or [35, Table 2]
Multiplication and Squaring	$\mathcal{O}(\log^2 x)$	$\mathcal{O}(x \log(x) \log(\log(x)))$	[59]

Table 2: Assumed cost of arithmetic operations in $U_{\mathrm{P}}^{\min}$. Each operations has input numbers of bit length x_i and outputs numbers of size x_{i+1}

Operation	Input bit lengths	T-Depth	GCost
1, parallel multiplication	$x_0 = \min(\mathfrak{b}, \xi)$	$\log^2(x_0)$	$h^2 \cdot x_0 \log(x_0) \log(\log(x_0))$
2, binary tree addition	$x_1 = \mathfrak{b} + \xi$	$\log h \cdot \log(x_1)$	$h^2 \cdot x_1$
3, squaring	$x_2 = x_1 + \log h$	$\log^2(x_2)$	$h \cdot x_2 \log(x_2) \log(\log(x_2))$
4, binary tree addition	$x_3 = 2\, x_2$	$\log h \cdot \log(x_3)$	$h \cdot x_3$
5, comparison	$x_4 = x_3 + \log h$	$\log(x_4)$	x_4

Depth and cost estimation of $\mathcal{W}$. In our setting, quantum enumeration is being performed on a tree $\mathcal{T}(g \in Z_k)$ of height $h = n - k$, where n is the dimension of the full lattice $\Lambda = \Lambda((b_1, \ldots, b_n))$ being enumerated. This process would require performing arithmetic using the projected lattice basis vectors $(\pi_{n-\ell+1}(b_{n-\ell+1}), \ldots, \pi_{n-\ell+1}(b_{n-k}))$ of Λ for all levels $k < \ell \leq n$. An operator $U_{\mathrm{P}}^{\min}$ is designed as to evaluate the predicate P which identifies projected vectors $v \in \pi_{n-\ell+1}(\Lambda)$ such that $||v|| \leq R_\ell$ and $\pi_{n-k+1}(v) = g$. In order to lower-bound the cost of operation, we only consider the case of evaluating this inequality at $\ell = n$, where $(\pi_{n-\ell+1}(b_{n-\ell+1}), \ldots, \pi_{n-\ell+1}(b_{n-k})) = (b_1, \ldots, b_h)$. Evaluating predicate P becomes checking whether $||\sum_{i \leq h} c_i b_i||^2 \leq R^2 - ||g||^2$ for some integer coefficients $(c_i)_i$. Since $(b_1, \ldots, b_h)$ span an h-dimensional vector space, we consider them to be h-dimensional by assuming an appropriate change of basis was applied.[6] Our estimate for the cost of $U_{\mathrm{P}}^{\min}$ is derived as in Tab. 2 applying the following sequence of operations:

1. Parallel multiplication of h^2 pairs $(c_i, (b_i)_j) \mapsto c_i (b_i)_j$, of $\mathfrak{b}$- and ξ-bit length, outputting numbers of bit length $\mathfrak{b} + \xi$.
2. Addition of coefficients $(c_1 (b_1)_j, \ldots, c_h (b_h)_j) \mapsto \sum_i c_i (b_i)_j$ for $j \in [h]$. These additions can be run in parallel over j. For a fixed j, the corresponding sum is run by adding terms in pairs, forming a binary tree of sums. Each $\sum_i c_i (b_i)_j$ output is $\mathfrak{b} + \xi + \log h$ bits long.
3. Squaring the $\sum_i c_i (b_i)_j$ sums in parallel (output bit length $2(\mathfrak{b} + \xi + \log h)$)

[6] In classical implementations, this computation benefits from extensive caching of Gram-Schmidt orthogonalisation operations and results [76]. Asymptotically, the number of individual arithmetic operations is the same as if computing directly from the basis $(b_1, \ldots, b_h)$.

Table 3: Kyber parameters [75, Sec 4.3] with corresponding BKZ block-sizes required for the primal attack. Column $\log \#\mathcal{T}^M$ reports our estimated lower bound on the number of nodes enumeration tree of dimension β if using extreme cylinder pruning with $M = 2^{64}$ and targeting success probability ≈ 1, following [12, Eq. (16)].

Scheme	LWE dimension n	Modulus q	Block-size β	$\log \#\mathcal{T}^M$	Target bit security
Kyber-512	512	3329	406	172.5	128
Kyber-768	768	3329	623	323.2	192
Kyber-1024	1024	3329	873	517.7	256

4. Adding the squared sums in a binary-tree fashion to obtain $||\sum_{i\leq h} c_i b_i||^2 = \sum_j (\sum_i c_i (b_i)_j)^2$ of bit length $2(\mathfrak{b} + \xi + \log h) + \log h$.
5. Comparison with the (adjusted) pruning bound $R^2 - ||g||^2$.

In this setting, for the purpose of estimating a minimum cost of the attack we consider the depth and gate-cost of $\mathcal{W}$ to correspond to the sum of depths and costs of the five operations above.

5 Estimating Quantum Enumeration Attacks on Kyber

In § 3 and 4, we have described methods to explore the enumeration tree under a MaxDepth limitation (cf. § 3.3 and 3.4), and introduced two different instantiations of the quantum backtracking operator $\mathcal{W}$. In this section, we leverage these results to present cost estimations for primal lattice reduction attacks using quantum enumeration against *Kyber* [75], the post-quantum key encapsulation mechanism selected by NIST for standardisation in 2022. To that end, we compute lower bounds on the cost of combined classical-quantum cylinder pruning in the gate-cost metric against the three different parametrisations of Kyber (cf. Tab. 3). For each of them, we consider attacks within MaxDepth $= 2^{40}, 2^{64}, 2^{96}$, as suggested by NIST [60], each assuming the two different instantiations of the operator $\mathcal{W}$ outlined in § 4.

5.1 Attack Setting

Our aim is to estimate a lower bound on the possible cost of classical-quantum enumeration in the setting of lattice-based cryptography. As a case-study, we look at the *primal attack* on Kyber, where a block lattice reduction algorithm is used to recover the secret key of the cryptosystem by reducing an embedding lattice [15] constructed using the public key.

We follow the convention of using BKZ as the lattice reduction algorithm, and assume that its shortest vector problem (SVP) oracle is instantiated using our classical-quantum enumeration approach. The common approach to primal attack estimates is to choose a *cost model* for BKZ that accounts for the cost

of running the SVP oracle and for the number of calls made [5]. Normally, cost models will use a closed formula for the cost of enumeration in dimension β to account for the cost of the SVP oracle, either fitted or derived from theory or experiments. This is then used with some estimation script such as the `lwe-estimator` [8], which will simulate the effect of lattice reduction and find the cheapest parametrisation of the attack leading to high success probability.

Since our setting involves an implicit relation between the gate-cost of the SVP oracle and the MaxDepth constraint, we don't attempt to fit our results on a curve as a function of β. Instead, we opt for calling an estimator script assuming the optimistic cost of classical enumeration obtained as part of our analysis (cf. App. A), which assumes that input bases achieve a linear lattice profile (as predicted by the Geometric Series Assumption, using the root-Hermite factor $((\pi\beta)^{1/\beta}\beta/(2\pi e))^{1/(2(\beta-1))}$ from [21]) so that enumeration costs $2^{\beta \log \beta/8+O(\beta)}$ (recently experimentally confirmed as possible in [3]), and assuming specifically the lower bound costs for extreme cylinder pruning proven in [12]. From this cost estimation we obtain three different block sizes β for the three parameter sets of Kyber, reported in Tab. 3. We then proceed to estimate the gate-cost of classical-quantum enumeration in dimension β under different MaxDepth values, and compare these with the corresponding bit security (e.g. 2^{128} for Kyber-512), approximate gate-cost of Grover search on AES for the corresponding category (e.g. Kyber-512 with AES-128), and a hypothetical asymptotic quadratic speedup for the full tree.

It is important to highlight an issue towards claiming lower bounds on the cost of classical-quantum enumeration, and how we address it. As pointed out in [13] and mentioned in § 3, the expected speedup of quantum enumeration over equivalent classical enumeration may be more than quadratic, depending on the probability distribution of the size of the trees being enumerated, due to Jensen's inequality implying $\mathbb{E}[\sqrt{\#\mathcal{T}}] \leq \sqrt{\mathbb{E}[\#\mathcal{T}]}$. Since we would like to provide lower bounds to the expected attack cost, we define $z \geq 0$ such that $\mathbb{E}[\sqrt{\#\mathcal{T}}] = 2^{-z}\sqrt{\mathbb{E}[\#\mathcal{T}]}$, and estimate the attack cost for $z = 0, \ldots, 64$. While we do not know what the value of z may be for lattices encountered in cryptanalysis,[7] this allows us to delegate the estimation of the concrete cost to future analysis on the distribution of $\#\mathcal{T}$, while clearly identifying *threshold values* z_0, such that $z \geq z_0$ may imply possible effective attacks, while $z < z_0$ would indicate that classical-quantum enumeration would not appear to obviously put the security of Kyber at risk.

We note that an alternative approach could be deriving a lower bound to the Jensen's gap, depending on some other parameter of the problem. We attempt this approach in App. G, where we derive bounds depending on the variance of $\#\mathcal{T}$. This however presents the same issue as above, that is that we are not aware of the exact distribution of $\#\mathcal{T}$, meaning that while it provides a different

[7] Albeit not at cryptographically relevant sizes, in App. C we present the results of small-dimension measurements of the Jensen gap against small q-ary lattices. We observe that for pruned enumeration in dimension $\beta \approx 60$, the gap appears to be around $z \approx 1$.

Table 4: Attack parameters for experimental evaluation.

$\textsc{MaxDepth} \in \{2^{40}, 2^{64}, 2^{96}\}$,	$y \in \{0, \ldots, 64\}$,	$z \in \{0, \ldots, 64\}$,	$M = 2^m \in \{2^0, 2^1, \ldots, 2^{64}\}$,	
$n \in \{406, 623, 873\}$,	$k \in [n]$,	$\mathrm{DF}(\cdot) = \mathrm{QD}(\cdot) = 1$,	$\mathrm{WQ}(\cdot) = \sqrt{\#\mathcal{T}n}$	

Table 5: Legend for plots reporting attack costs under MaxDepth constraint.

2^y	# subtrees rooted at level k combined under a single FindMV call, cf. § 3.3	QRACM	Maximum amount of quantum accessible classical memory available, a constraint on 2^y
Exp. cost of Grover on AES	Expected GCost of Grover search for AES key recovery [39, Tab. 12] with prob. ≈ 1	Quantum GCost	Expected combined cost (cf. Eq. (6)) of all quantum circuits that enumerate the level below k
MaxDepth	Constraint on T-Depth(QPE($\mathcal{W}$)), cf. Eq. (7)	2^z	Multiplicative Jensen's Gap, cf. Def. 1
Classical GCost	Expected # nodes (cf. Eq. (4)) enumerated classically up to level k	Quasi-Sqrt (class. cost)	Asymptotic runtime of quantum enumeration, $\approx \left(2^y \cdot N^M_{k,n-k} \cdot (n-k)\right)^{1/2}$
Targeted security	2^{128} (resp., 2^{192}, 2^{256}) for Kyber-512 (resp., -768, -1024)	Exp. cost of classical enum.	Lower bounds on the cost of enumeration with extreme cilinder pruning [12]
Total GCost	Classical cost + quantum cost	$M = 2^m$	Number of pruned trees enumerated

formulation of the problem, it currently does not represent a better alternative to testing many values of z looking for threshold values.

Overall, our estimation code for the cost of enumeration on a β-dimensional lattice bases is given in input a multiplicative Jensen gap 2^z and a MaxDepth constraint, and optimizes the number of combined bases M, the level k which separates classical and quantum enumeration as well as the maximal number 2^y of combined nodes on this level, looking for the cheapest possible attack. We estimate the cost of extreme pruning attacks with success probability ≈ 1. We give an overview over all parameters used in the estimation process in Tab. 4. We have made our source code used to produce or experimental results, tables, and plots publicly available at https://github.com/mtiepelt/QuantumLatticeEnumeration.

5.2 Cost Estimation of the Attack

Cost metrics and success conditions. As mentioned in § 2.3, the cost of a quantum algorithm can be measured using various metrics. In this paper, we prefer to focus on the number of classical and quantum gates required by the attack in total, since, plausibly, applying one quantum gate requires running one classical computation on some microcontroller [41], meaning that to some extent these two quantities can be compared and combined. In these terms, one could say a combined classical-quantum enumeration attack was successful if the total

number of gates required[8] was lower than some threshold capturing some notion of security.

The success of an attack can be defined in multiple ways. For a scheme like Kyber-512 (with analogous notions for -768 and -1024), one could consider an attack successful if its cost is lower than $2^{\text{target bit security}} = 2^{128}$. Alternatively, one can note that submissions to the NIST standardisation process, such as Kyber, were required to propose parameters for cryptographic primitives as hard to break as AES or SHA (depending on the targeted security category). This would imply a notion where a quantum attack against Kyber may be considered successful only if its cost is lower than the number of gates required to run Grover search against AES, which we estimate using Tables 10 and 12 of [39].[9] It should be noticed that the reason such a separation between classical and quantum attacks is due to the assumed and yet-unknown hidden costs of quantum computation. To be conservative, we compare the security notions above to a simple sum of classical and quantum gate costs, which is likely extremely generous towards quantum computation.

Another interesting threshold check is whether the attack cost under depth constraints ever achieves a "quasi-quadratic" speedup over classical enumeration (meaning going from $\#\mathcal{T}$ gates to $\sqrt{\#\mathcal{T} \cdot h}$, where h is the height of $\mathcal{T}$), as it could be expected from Thm. 1.

In the following paragraphs we proceed to explore whether these thresholds could be *plausibly* met for classical-quantum enumeration with cylinder pruning, where plausibility depends on the multiplicative Jensen gap required (cf. Def. 1), if any, to achieve them. An overview of our results, based on the strict lower bounds for DF, QD, and WQ stated in § 3.3, can be found in Tab. 6. A similar table based on our estimates for these quantities performed in App. F can be found in App. F.1.

Cost estimation without MaxDepth restrictions. We start by estimating the cost of enumeration without MaxDepth restrictions—the most favourable setting to the adversary. In this setting, a quadratic speedup in terms of quantum depth can be achieved as the full enumeration tree can be enumerated directly within a call to FindMV, meaning no classical phase is required. This means, the attack consists of calling FindMV($\mathcal{T}^M$) once. No QRACM is needed, and the classical cost is null. The dependency between the cost of the attack and

[8] We lower bound this as the number of nodes visited in the classical phase of the attack plus the number of quantum gates applied during the quantum phase of the attack.

[9] Under no MaxDepth constraint, [39, Table 10] suggests that the GCost of key recovery against AES-128 (resp. AES-192, AES-256) with success probability ≈ 1 is $\approx 2^{83}$ (resp. 2^{115}, 2^{148}). Under a depth constraint, [39, Table 12] suggests the GCost $\approx 2^{157}$/MaxDepth (resp. 2^{221}/MaxDepth, 2^{285}/MaxDepth). We note that further improvements in the design of the Grover oracles against AES have achieved minor speedups in terms of overall gate cost.

Table 6: Summary of the values for the Jensen's gap 2^z at crossover points of our combined classical-quantum enumeration attacks against Kyber and different "success metrics" to claim an attack. We remark that exact crossovers happen at fractional values of z. In this table we round down threshold values of z. The cost of Grover's search against AES is taken from [39, Tables 10 and 12].

more likely to be feasible ← → less likely to be feasible

MD	Kyber	$\log \mathbb{E}[\text{GCost}]$ (with $\mathcal{W}$ as in § 4.1) below... Target security	Grover on $\text{AES}_{\{128,192,256\}}$	Quasi-Sqrt $\sqrt[1/b]{\#\mathcal{T} \cdot h}$	$\log \mathbb{E}[\text{GCost}]$ (with $\mathcal{W}$ as in § 4.2) below... Target security	Grover on $\text{AES}_{\{128,192,256\}}$	Quasi-Sqrt $\sqrt[1/b]{\#\mathcal{T} \cdot h}$
2^{40}	-512	$z \geq 7,\ k \leq 92$	$z \geq 13,\ k \leq 83$	$z \geq 26,\ k \leq 59$	$z \geq 23,\ k \leq 96$	$z \geq 29,\ k \leq 79$	$z \geq 42,\ k \leq 63$
	-768	$z \geq 51,\ k \leq 114$	$z \geq 57,\ k \leq 106$	$z \geq 64,\ k \leq 77$	$z > 64$	$z > 64$	$z > 64$
	-1024	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$
2^{64}	-768	$z \geq 39,\ k \leq 114$	$z \geq 57,\ k \leq 77$	$z \geq 52,\ k \leq 77$	$z \geq 55,\ k \leq 106$	$z > 64$	$z > 64$
	-512	$z \geq 0,\ k \leq 83$	$z \geq 13,\ k \leq 64$	$z \geq 14,\ k \leq 59$	$z \geq 11,\ k \leq 79$	$z \geq 29,\ k \leq 54$	$z \geq 30,\ k \leq 46$
	-1024	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$
2^{96}	-512	$z \geq 0,\ k \leq 46$	$z \geq 8,\ k \leq 46$	$z \geq 1,\ k \leq 46$	$z \geq 0,\ k \leq 46$	$z \geq 33,\ k \leq 46$	$z \geq 25,\ k \leq 46$
	-768	$z \geq 23,\ k \leq 77$	$z \geq 56,\ k \leq 44$	$z \geq 36,\ k \leq 77$	$z \geq 40,\ k \leq 77$	$z > 64$	$z \geq 53,\ k \leq 62$
	-1024	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$
∞	-512	$z \geq 0,\ k = 0$	$z \geq 9,\ k = 0$	$z \geq 1,\ k = 0$	$z \geq 0,\ k = 0$	$z \geq 33,\ k = 0$	$z \geq 26,\ k = 0$
	-768	$z \geq 0,\ k = 0$	$z \geq 52,\ k = 0$	$z \geq 1,\ k = 0$	$z \geq 1,\ k = 0$	$z > 64$	$z \geq 27,\ k = 0$
	-1024	$z \geq 9,\ k = 0$	$z > 64$	$z \geq 1,\ k = 0$	$z \geq 35,\ k = 0$	$z > 64$	$z \geq 28,\ k = 0$

the Jensen gap 2^z is straightforward in this setting, with the quantum cost exponentially reducing as z increases.

We report our results in Tab. 6, under the "MD = ∞" rows and make a few observations next. First, quadratic speedups can be achieved in the very conservative query-based model of § 4.1, but seem already less likely under the mild assumptions of the circuit-based model of $\mathcal{W}$ in § 4.2, which appear to require $z \gtrapprox 26$. Second, achieving a gate cost under the target bit-security (e.g., attacking Kyber-768 in less than 2^{192} quantum gates) seems plausible in most cases. Yet, this is a crude notion of security in the quantum-attack setting. Indeed, comparing the attack cost with that of Grover on AES gives a picture suggesting that quantum enumeration-powered primal lattice attacks on Kyber may be unlikely to succeed. Only Kyber-512 appears to be plausibly approachable, $z \geq 0$ sufficing, yet this is still only considering the query model for $\mathcal{W}$. A larger Jensen gap of $z \geq 33$ is required in the circuit-based model of $\mathcal{W}$ in § 4.2.

Cost estimation with MaxDepth restrictions. We now consider the effect of depth restrictions on the cost of the attack. In most cases, depth restrictions mean that we will need to use a combined classical-quantum attack as described in § 3.2 and 3.3, where classical enumeration is run up to level k, as to create subsets $\{g_1, \ldots, g_{2^y}\} \subset Z_k$. A 'virtual' root node g is added as 'parent' of these,

and quantum enumeration is run on the resulting tree $\mathcal{T}(g)$. This process requires about 2^y QRACM to store the $\{g_i\}_{i\leq 2^y}$, and is run on the extreme cylinder pruning enumeration tree $\mathcal{T}^M$ from § 3.4.

To compute Eq. (6) given a Jensen gap 2^z, we minimize the total cost of the combined classical-quantum enumeration with extreme cylinder pruning and success probability $\approx$ 1 over: the number $M = 2^m$ of re-randomized bases used for extreme pruning, for $m \leq 64$; the amount 2^y of QRACM used; the level k at which classical enumeration ends and quantum enumeration starts, for $k \leq n$. For each set of parameters, we only consider those such that the depth of the quantum phase estimation $\mathrm{QPE}(\mathcal{W})$ (cf. Eq. (7)) is no larger than MaxDepth. For every $z \in \{0, ..., 64\}$, we output (m, y, k) minimizing the total cost. We report our results in Tab. 6, under the "MD $= 2^{40}, \ldots, 2^{96}$" rows.

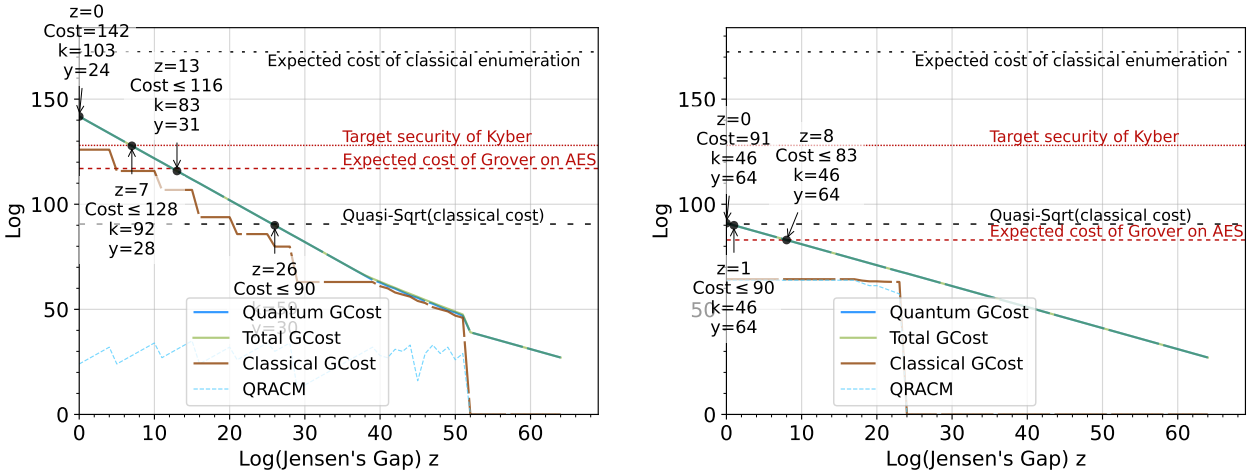
Compared to the unbounded-depth setting, we immediately notice that quadratic speedups appear unlikely to happen for Kyber-768 and -1024 already in the query-based model, § 4.1, and also for Kyber-512 in the circuit-based model, § 4.2, albeit keeping in mind the caveat that we do not known with certainty how likely a Jensen gap of, say, $z \geq 25$ is. As it is to be expected, the required gap diminishes as MaxDepth increases. Unlike in the unbounded-depth setting, achieving a total gate-cost below the target security value appears also unlikely, except in the case of Kyber-512, where some attack settings achieve this already at $z = 0$. As per the attack costing less than Grover search on AES, the situation is similar, with more stringent requirements for the Jensen gap. While in the query-based model attacks on Kyber-512 appear potentially more likely (c.f. $z \geq 8$ at MaxDepth $= 2^{96}$ or $z \geq 13$ at MaxDepth $= 2^{40}$), attacks on Kyber-768 and -1024 appear less plausible. Yet, moving away from the query-based model into the circuit-based model of $\mathcal{W}$, immediately ups the requirements of a successful attack on Kyber-512 up to $z \geq 29$.

We remark that whenever we can find a value of $z < 64$ such that a success target is achieved, the value of k is well below $n/2$. This matches our intuition: at $k \approx n/2$, the classical cost of the component would essentially equal the cost of the fully-classical attack, but would also incur into a quantum overhead.

An important difference between the bounded- and unbounded-depth settings for combined classical-quantum enumeration is the dependency of the total cost on the Jensen's gap, 2^z. Indeed, while in the unbounded setting the cost of the attack is simply proportional to 2^{-z}, in the bounded setting different values of MaxDepth and z imply different amounts of classical precomputation.

Since we do not have a clear prediction of the exact value of z for different enumeration tree distributions, we investigate how sensitive is the total cost of the attack to small changes in z by plotting the predicted classical and quantum gate costs and QRACM requirements as a function of z. In Fig. 4 we show the resulting plots for Kyber-512 at MaxDepth $= 2^{40}$ and 2^{96} in the query-based model, as a representative example. Plots for MaxDepth $= 2^{64}$ and for Kyber-768 and -1024, and for circuit-based model can be found in App. E.

Overall costs appear to decrease smoothly as z increases, without sudden reductions. Two peculiar phenomenons can be observed: the optimal attack is



(a) with MaxDepth $= 2^{40}$

(b) with MaxDepth $= 2^{96}$

Figure 4: Cost estimation for Kyber-512 under MaxDepth restrictions with the instantiation for operator $\mathcal{W}$ as in § 4.1. Cf. Tab. 5 for an expanded legend.

not achieved when the two phases of the attack are balanced, and the QRACM requirements show a saw-tooth behaviour, apparently related to the reduction of costs of the classical precomputation phase. We will now elaborate on these.

Unbalanced classical and quantum cost. The only parameter determining the classical cost is k. Since in our observed case the classical cost is always smaller than the quantum one, the only option for balancing the two would be by increasing k. Increasing $k \mapsto k+1$, the classical cost increases by an additive term $H_{k+1}/2$, while the quantum cost and the quantum depth approximately change by a factor $\sqrt{\frac{n-k-1}{n-k}\frac{H_{k+1}}{H_k}}$ and $\sqrt{\frac{n-k-1}{n-k}\frac{H_k}{H_{k+1}}}$ respectively. How overall this affects the quantum cost will depend on whether $H_{k+1}/H_k \gtrless 1$, as well as whether a different value of y is chosen as to keep using exactly MaxDepth quantum depth during the attack. From Tab. 6, it appears that the optimal attacks we find are in the $k < n/2$ regime, where $H_{k+1}/H_k > 1$ [32], meaning increasing k may increase both classical and quantum costs, which is undesirable. Due to the complexity of an analytic analysis, we believe the safer approach is looking for the optimal attack computationally. It would then appear that the lowest *classical plus quantum* cost is achieved with unbalanced quantum and classical costs, within the constraints we consider.

Saw-tooth QRACM. Suppose we are given values z, k and y corresponding to some point on the graphs in Figure 4. Recalling Eq. (7) and using the query model as an example, the depth constraint becomes $\frac{1}{2^z}\sqrt{(n-k)\cdot 2^y \cdot \mathbb{E}[\#\mathcal{T}(g \in Z_k)]} \approx$ MaxDepth. Suppose we increase z while always enforcing the depth constraint,

this implies an increase in $\sqrt{(n-k)\cdot 2^y \cdot \mathbb{E}[\#\mathcal{T}(g \in Z_k)]}$. This means our costing loop will explore increasing y and decreasing k (hence increasing $(n-k)\cdot \mathbb{E}[\#\mathcal{T}(g \in Z_k)]$). Increasing y always leaves the classical cost intact, and always reduces the quantum cost by a factor $\sqrt{2^y}$, at the cost of increasing the QRACM consumption. Decreasing k in the $k < n/2$ regime will increase $(n-k)$ and $\mathbb{E}[\#\mathcal{T}]$ and decrease H_k, meaning the classical cost will decrease and the quantum cost may increase or decrease (cf. Eq. (6)). In practice, the complicated trade-off landscape appears (from our calculations) to result in having k stay the same and y increase most of the time, with k decreasing sporadically. Overall this results in the classical cost being a decreasing step function and the QRACM be "piece-wise increasing" sawtooth function, with peaks of the sawtooth preceeding downward steps of the classical cost function, which correspond to points where z increases enough that decreasing k is beneficial.

Conclusions. From this overview of our cost estimations, one could be tempted to take home the message that quantum enumeration using cylinder pruning seems unlikely to lead to a break of Kyber-768 and Kyber-1024, but that Kyber-512 could potentially be exposed to such attacks to some extent. This, however, should not be necessarily surprising or particularly alarming. Kyber-512 was designed to be in the same security regime as AES-128, a cipher for which quadratic quantum speedups are in principle not dismissible, even in the case of bounded depth attacks. Furthermore, we would like to stress again that our estimates are extremely conservative: we attempt to use lower bounds everywhere we can, such as for the size of the enumeration trees and subtrees, the number of quantum phase estimations performed, the cost of QRACM and qubits (that we entirely ignore), the size of the quantum circuit $\mathcal{W}$ performing floating point arithmetic to compute lattice point projections in superposition. Moreover, it is not clear that a Jensen gap of $z \approx 10$ would be easy or likely to be achieved for BKZ-induced enumeration lattices. Keeping all of this in mind, it is hard to claim that Kyber-512 is at risk of being less secure than AES-128, when attacks on it in the circuit-based model of $\mathcal{W}$ require $z \approx 30$.

Instead, we believe that the take-home message of this case-study should be that imposing MaxDepth limitations to quantum backtracking appears to present a significant obstacle towards leveraging this technique for lattice cryptanalysis. Even accounting for the possible uncertainty on the value of the Jensen gap 2^z, it should be noted that our results seem to suggest that the cost of the attack varies smoothly as a function of z, and that future "ballpark" estimations of the gap could already provide some guarantees on the hardness of the attack, since no sudden drops in the cost of the attack appear to happen up to $z \leq 64$.

Open directions. Two clear directions for further research are open: the better estimations of the Jensen's Gap, and the analysis of quantum enumeration for other forms of pruning. A possible approach to the first problem is outlined in App. G. The second problem also appears of particular interest. Indeed, it could potentially be the case that while quantum enumeration does not prove effect-

ive in limited depth settings using cylinder pruning, other pruning techniques, possibly designed *ad hoc* to this setting, could be more viable.

Acknowledgements. We would like to thank Sam Jaques for multiple fruitful conversations, Joe Rowell for help understanding the source code of `fplll`, and Martin R. Albrecht for offering computing power for running our experiments.

References

1. Aaronson, S., Ambainis, A.: Quantum search of spatial regions. pp. 200–209 (2003). https://doi.org/10.1109/SFCS.2003.1238194
2. Ajtai, M., Kumar, R., Sivakumar, D.: A sieve algorithm for the shortest lattice vector problem. pp. 601–610 (2001). https://doi.org/10.1145/380752.380857
3. Albrecht, M.R., Bai, S., Fouque, P.A., Kirchner, P., Stehlé, D., Wen, W.: Faster enumeration-based lattice reduction: Root hermite factor $k^{1/(2k)}$ time $k^{k/8+o(k)}$. pp. 186–212 (2020). https://doi.org/10.1007/978-3-030-56880-1_7
4. Albrecht, M.R., Bai, S., Li, J., Rowell, J.: Lattice reduction with approximate enumeration oracles - practical algorithms and concrete performance. pp. 732–759 (2021). https://doi.org/10.1007/978-3-030-84245-1_25
5. Albrecht, M.R., Curtis, B.R., Deo, A., Davidson, A., Player, R., Postlethwaite, E.W., Virdia, F., Wunderer, T.: Estimate all the LWE, NTRU schemes! pp. 351–367 (2018). https://doi.org/10.1007/978-3-319-98113-0_19
6. Albrecht, M.R., Ducas, L., Herold, G., Kirshanova, E., Postlethwaite, E.W., Stevens, M.: The general sieve kernel and new records in lattice reduction. pp. 717–746 (2019). https://doi.org/10.1007/978-3-030-17656-3_25
7. Albrecht, M.R., Gheorghiu, V., Postlethwaite, E.W., Schanck, J.M.: Estimating quantum speedups for lattice sieves. pp. 583–613 (2020). https://doi.org/10.1007/978-3-030-64834-3_20
8. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology **9**(3), 169–203 (2015). https://doi.org/doi:10.1515/jmc-2015-0016, https://doi.org/10.1515/jmc-2015-0016
9. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange - A new hope. pp. 327–343 (2016)
10. Ambainis, A., Kokainis, M.: Quantum algorithm for tree size estimation, with applications to backtracking and 2-player games. In: Hatami, H., McKenzie, P., King, V. (eds.) Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017. pp. 989–1002. ACM (2017). https://doi.org/10.1145/3055399.3055444, https://doi.org/10.1145/3055399.3055444
11. Aono, Y., Espitau, T., Nguyen, P.Q.: Random lattices: Theory and practice, preprint, available at https://espitau.github.io/bin/random_lattice.pdf
12. Aono, Y., Nguyen, P.Q., Seito, T., Shikata, J.: Lower bounds on lattice enumeration with extreme pruning. pp. 608–637 (2018). https://doi.org/10.1007/978-3-319-96881-0_21
13. Aono, Y., Nguyen, P.Q., Shen, Y.: Quantum lattice enumeration and tweaking discrete pruning. pp. 405–434 (2018). https://doi.org/10.1007/978-3-030-03326-2_14

14. Aono, Y., Wang, Y., Hayashi, T., Takagi, T.: Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator. pp. 789–819 (2016). https://doi.org/10.1007/978-3-662-49890-3_30
15. Bai, S., Galbraith, S.D.: Lattice decoding attacks on binary lwe. In: Information Security and Privacy: 19th Australasian Conference, ACISP 2014, Wollongong, NSW, Australia, July 7-9, 2014. Proceedings 19. pp. 322–337. Springer (2014)
16. Becker, A., Ducas, L., Gama, N., Laarhoven, T.: New directions in nearest neighbor searching with applications to lattice sieving. In: Krauthgamer, R. (ed.) Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016. pp. 10–24. SIAM (2016). https://doi.org/10.1137/1.9781611974331.ch2, https://doi.org/10.1137/1.9781611974331.ch2
17. Bessen, A.J.: Lower bound for quantum phase estimation. Physical Review A **71**(4), 042313 (2005)
18. Bonnetain, X., Chailloux, A., Schrottenloher, A., Shen, Y.: Finding many collisions via reusable quantum walks - application to lattice sieving. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V. Lecture Notes in Computer Science, vol. 14008, pp. 221–251. Springer (2023). https://doi.org/10.1007/978-3-031-30589-4_8, https://doi.org/10.1007/978-3-031-30589-4_8
19. Campbell, E., Khurana, A., Montanaro, A.: Applying quantum algorithms to constraint satisfaction problems. Quantum **3**, 167 (jul 2019). https://doi.org/10.22331/q-2019-07-18-167, https://doi.org/10.22331%2Fq-2019-07-18-167
20. Chailloux, A., Loyer, J.: Lattice sieving via quantum random walks (2021)
21. Chen, Y.: Réduction de réseau et sécurité concrète du chiffrement complètement homomorphe. Ph.D. thesis, Paris 7 (2013), https://archive.org/details/PhDChen13, available at https://archive.org/details/PhDChen13
22. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates. pp. 1–20 (2011). https://doi.org/10.1007/978-3-642-25385-0_1
23. Dagdelen, Ö., Schneider, M.: Parallel enumeration of shortest lattice vectors. Cryptology ePrint Archive, Report 2010/097 (2010), https://eprint.iacr.org/2010/097
24. Davenport, J.H., Pring, B.: Improvements to quantum search techniques for block-ciphers, with applications to AES. pp. 360–384 (2020). https://doi.org/10.1007/978-3-030-81652-0_14
25. Detrey, J., Hanrot, G., Pujol, X., Stehlé, D.: Accelerating lattice reduction with FPGAs. pp. 124–143 (2010)
26. Draper, T.G., Kutin, S.A., Rains, E.M., Svore, K.M.: A logarithmic-depth quantum carry-lookahead adder. Quantum Info. Comput. **6**(4), 351–369 (jul 2006)
27. Elandt-Johnson, R.C., Johnson, N.L.: Survival models and data analysis. Wiley Series in Probability and Statistics, John Wiley & Sons, Nashville, TN (Sep 1980)
28. Fincke, U., Pohst, M.: Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. Mathematics of computation **44**(170), 463–471 (1985)
29. Fincke, U., Pohst, M.E.: Improved methods for calculating vectors of short length in a lattice. Mathematics of Computation (1985)
30. Fowler, A.G., Mariantoni, M., Martinis, J.M., Cleland, A.N.: Surface codes: Towards practical large-scale quantum computation. Physical Review A **86**(3) (sep 2012). https://doi.org/10.1103/physreva.86.032324, https://doi.org/10.1103%2Fphysreva.86.032324

31. Gama, N., Nguyen, P.Q.: Finding short lattice vectors within Mordell's inequality. pp. 207–216 (2008). https://doi.org/10.1145/1374376.1374408
32. Gama, N., Nguyen, P.Q., Regev, O.: Lattice enumeration using extreme pruning. pp. 257–278 (2010). https://doi.org/10.1007/978-3-642-13190-5_13
33. Gao, X., Sitharam, M., Roitberg, A.E.: Bounds on the jensen gap, and implications for mean-concentrated distributions. Australian Journal of Mathematical Analysis and Applications **16**(2), 1–16 (2019), https://ajmaa.org/cgi-bin/paper.pl?string=v16n2/V16I2P14.tex
34. Giovannetti, V., Lloyd, S., Maccone, L.: Quantum random access memory. Physical review letters **100**(16), 160501 (2008)
35. Häner, T., Jaques, S., Naehrig, M., Roetteler, M., Soeken, M.: Improved quantum circuits for elliptic curve discrete logarithms. In: Post-Quantum Cryptography: 11th International Conference, PQCrypto 2020, Paris, France, April 15–17, 2020, Proceedings 11. pp. 425–444. Springer (2020)
36. Häner, T., Roetteler, M., Svore, K.M.: Factoring using 2n + 2 qubits with toffoli based modular multiplication. Quantum Info. Comput. **17**(7–8), 673–684 (jun 2017)
37. Hanrot, G., Pujol, X., Stehlé, D.: Analyzing blockwise lattice algorithms using dynamical systems. pp. 447–464 (2011). https://doi.org/10.1007/978-3-642-22792-9_25
38. Hermans, J., Schneider, M., Buchmann, J., Vercauteren, F., Preneel, B.: Parallel shortest lattice vector enumeration on graphics cards. pp. 52–68 (2010)
39. Jaques, S., Naehrig, M., Roetteler, M., Virdia, F.: Implementing grover oracles for quantum key search on AES and LowMC. pp. 280–310 (2020). https://doi.org/10.1007/978-3-030-45724-2_10
40. Jaques, S., Rattew, A.G.: Qram: A survey and critique (2023)
41. Jaques, S., Schanck, J.M.: Quantum cryptanalysis in the RAM model: Claw-finding attacks on SIKE. pp. 32–61 (2019). https://doi.org/10.1007/978-3-030-26948-7_2
42. Jones, N., Van Meter, R., Fowler, A., McMahon, P., Kim, J., Ladd, T., Yamamoto, Y.: Layered architecture for quantum computing. Physical Review X **2** (10 2010). https://doi.org/10.1103/PhysRevX.2.031007
43. Kannan, R.: Improved algorithms for integer programming and related lattice problems. pp. 193–206 (1983). https://doi.org/10.1145/800061.808749
44. Kirshanova, E., Mårtensson, E., Postlethwaite, E.W., Moulik, S.R.: Quantum algorithms for the approximate k-list problem and their application to lattice sieving. pp. 521–551 (2019). https://doi.org/10.1007/978-3-030-34578-5_19
45. Koch, D., Samodurov, M., Projansky, A., Alsing, P.M.: Gate-based circuit designs for quantum adder-inspired quantum random walks on superconducting qubits. International Journal of Quantum Information **20**(03), 2150043 (2022)
46. Kuo, P.C., Schneider, M., Dagdelen, Ö., Reichelt, J., Buchmann, J., Cheng, C.M., Yang, B.Y.: Extreme enumeration on GPU and in clouds - - how many dollars you need to break SVP challenges -. pp. 176–191 (2011). https://doi.org/10.1007/978-3-642-23951-9_12
47. Kuperberg, G.: Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem (2011), arXiv preprint arXiv:1112.3333
48. Laarhoven, T.: Sieving for shortest vectors in lattices using angular locality-sensitive hashing. pp. 3–22 (2015). https://doi.org/10.1007/978-3-662-47989-6_1
49. Laarhoven, T., Mosca, M., van de Pol, J.: Solving the shortest vector problem in lattices faster using quantum search. pp. 83–101 (2013). https://doi.org/10.1007/978-3-642-38616-9_6

50. Li, J., Nguyen, P.Q.: A complete analysis of the BKZ lattice reduction algorithm. Cryptology ePrint Archive, Report 2020/1237 (2020), https://eprint.iacr.org/2020/1237
51. Lindner, R., Peikert, C.: Better key sizes (and attacks) for LWE-based encryption. pp. 319–339 (2011). https://doi.org/10.1007/978-3-642-19074-2_21
52. Lyubashevsky, V., Ducas, L., Kiltz, E., Lepoint, T., Schwabe, P., Seiler, G., Stehlé, D., Bai, S.: CRYSTALS-DILITHIUM. Tech. rep., National Institute of Standards and Technology (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022
53. Micciancio, D., Walter, M.: Fast lattice point enumeration with minimal overhead. pp. 276–294 (2015). https://doi.org/10.1137/1.9781611973730.21
54. Micciancio, D., Walter, M.: Practical, predictable lattice basis reduction. pp. 820–849 (2016). https://doi.org/10.1007/978-3-662-49890-3_31
55. Montanaro, A.: Quantum-walk speedup of backtracking algorithms. Theory Comput. **14**(1), 1–24 (2018). https://doi.org/10.4086/toc.2018.v014a015, https://doi.org/10.4086/toc.2018.v014a015
56. Montanaro, A.: Quantum-walk speedup of backtracking algorithms. Theory of Computing **14**(15), 1–24 (2018). https://doi.org/10.4086/toc.2018.v014a015, https://theoryofcomputing.org/articles/v014a015
57. Muñoz-Coreas, E., Thapliyal, H.: Quantum circuit design of a t-count optimized integer multiplier. IEEE Transactions on Computers **68**(5), 729–739 (2019). https://doi.org/10.1109/TC.2018.2882774
58. Nguyen, P.Q., Vidick, T.: Sieve algorithms for the shortest vector problem are practical. J. Math. Cryptol. **2**(2), 181–207 (2008). https://doi.org/10.1515/JMC.2008.009, https://doi.org/10.1515/JMC.2008.009
59. Nie, J., Zhu, Q., Li, M., Sun, X.: Quantum circuit design for integer multiplication based on schönhage-strassen algorithm. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems pp. 1–1 (2023). https://doi.org/10.1109/TCAD.2023.3279300
60. NIST: Submission requirements and evaluation criteria for the post-quantum cryptography standardization process (2016), https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf
61. Pham, P., Svore, K.M.: A 2d nearest-neighbor quantum architecture for factoring in polylogarithmic depth. Quantum Inf. Comput. **13**(11-12), 937–962 (2013)
62. Pohst, M.: On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications. SIGSAM Bull. **15**(1), 37–44 (feb 1981). https://doi.org/10.1145/1089242.1089247, https://doi.org/10.1145/1089242.1089247
63. Preskill, J.: Quantum Computing in the NISQ era and beyond. Quantum **2**, 79 (Aug 2018). https://doi.org/10.22331/q-2018-08-06-79, https://doi.org/10.22331/q-2018-08-06-79
64. Prest, T., Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: FALCON. Tech. rep., National Institute of Standards and Technology (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022
65. Pujol, X., Stehlé, D.: Rigorous and efficient short lattice vectors enumeration. pp. 390–405 (2008). https://doi.org/10.1007/978-3-540-89255-7_24
66. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) Proceedings of the 37th Annual

ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005. pp. 84–93. ACM (2005). https://doi.org/10.1145/1060590.1060603, https://doi.org/10.1145/1060590.1060603
67. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM **56**(6), 34:1–34:40 (2009). https://doi.org/10.1145/1568318.1568324, https://doi.org/10.1145/1568318.1568324
68. Rudin, W.: Real and Complex Analysis. McGraw-Hill series in higher mathematics, McGraw-Hill Professional, New York, NY, 3 edn. (Sep 1986)
69. Ruiz-Perez, L., Garcia-Escartin, J.C.: Quantum arithmetic with the quantum fourier transform. Quantum Information Processing **16**, 1–14 (2017)
70. Schanck, J.M., Hulsing, A., Rijneveld, J., Schwabe, P.: NTRU-HRSS-KEM. Tech. rep., National Institute of Standards and Technology (2017), available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-1-submissions
71. Schnorr, C., Euchner, M.: Lattice basis reduction: Improved practical algorithms and solving subset sum problems. Mathematical Programming **66**, 181–199 (08 1994). https://doi.org/10.1007/BF01581144
72. Schnorr, C.: Lattice reduction by random sampling and birthday methods. In: Alt, H., Habib, M. (eds.) STACS 2003, 20th Annual Symposium on Theoretical Aspects of Computer Science, Berlin, Germany, February 27 - March 1, 2003, Proceedings. Lecture Notes in Computer Science, vol. 2607, pp. 145–156. Springer (2003). https://doi.org/10.1007/3-540-36494-3_14, http://dx.doi.org/10.1007/3-540-36494-3_14
73. Schnorr, C.P., Euchner, M.: Lattice basis reduction: Improved practical algorithms and solving subset sum problems. In: FCT (1991)
74. Schnorr, C.P., Euchner, M.: Lattice basis reduction: Improved practical algorithms and solving subset sum problems. Mathematical Programming (1994)
75. Schwabe, P., Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Seiler, G., Stehlé, D., Ding, J.: CRYSTALS-KYBER. Tech. rep., National Institute of Standards and Technology (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022
76. development team, T.F.: fplll, a lattice reduction library, Version: 5.4.2 (2023), https://github.com/fplll/fplll, available at https://github.com/fplll/fplll
77. The Sage Developers: SageMath, the Sage Mathematics Software System (Version 9.3) (2021), `https://www.sagemath.org`
78. Zalka, C.: Grover's quantum searching algorithm is optimal. Physical Review A **60**(4), 2746 (1999)
79. Zou, J., Wei, Z., Sun, S., Liu, X., Wu, W.: Quantum circuit implementations of AES with fewer qubits. pp. 697–726 (2020). https://doi.org/10.1007/978-3-030-64834-3_24

A Estimating the Size of Lattice Enumeration Trees

In this appendix we perform some standard computations on the size of enumeration trees. We use these to compute the value of H_k in § 3 and 5. This analysis closely follows [32].

Size of the enumeration tree. The cost of lattice enumeration is typically estimated to be equal to the number of nodes visited by the algorithm. These nodes form the "enumeration tree". If we let Z_k be the set of nodes on the k^{th} level of the tree (that is, at distance k from the root node), H_k be the expected cardinality of Z_k for the enumeration tree of a random lattice basis and N the expected total number of nodes in the tree visited by the algorithm, we have

$$N \coloneqq \mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}[\#\mathcal{T}] = \frac{1}{2}\sum_{k=1}^{n} \mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}[|Z_k|] = \frac{1}{2}\sum_{k=1}^{n} H_k,$$

where the $1/2$ factor is due to exploiting lattices' additive symmetry to avoid visiting half of the tree. The cardinality of Z_k depends on the pruning strategy and on the geometry of the projective sublattices implied by the lattice bases, and can be estimated using the Gaussian heuristic. For example, we have

$$Z_k = \begin{cases} \mathrm{Ball}_k(\vec{0}, R) \cap \pi_{n-k+1}(\Lambda), & \text{using no pruning,} \\ C_{R_1,\ldots,R_k} \cap \pi_{n-k+1}(\Lambda), & \text{using cylinder pruning,} \end{cases}$$

where $C_{R_1,\ldots\ R_k} \coloneqq \{(x_1,\ldots,x_k) \in \mathbb{R}^k \mid \sum_{i\le j} x_i^2 \le R_j^2 \text{ for all } j \le k\}$ is the intersection of cylinders of increasing dimension and non-decreasing radius, and where the covolume of the projected sublattices $\pi_{n-k+1}(\Lambda)$ depends on the quality of the input lattice basis $B = (b_1,\ldots,b_n)$, with $\mathrm{covol}(\pi_{n-k+1}(\Lambda)) = \prod_{i=n-k+1}^{n} ||b_i^*|| = \mathrm{covol}(\Lambda)/\prod_{i=1}^{n-k} ||b_i^*||$. where $B^* = (b_1^*,\ldots,b_n^*)$ is the Gram-Schmidt orthogonalisation of B.

Cost estimation for the algorithm then reduces to estimating H_k, as done in [32,12], noting that $\log N \le \log(n \cdot \max_k H_k/2) \in O(\log \max_k H_k)$. We proceed to illustrate how to estimate H_k in the case of non-pruned enumeration. We will use these results, together with the analysis of [12], to numerically estimate lower bounds for H_k in the case of cylinder pruning.

A.1 Computing H_k when no Pruning is Used

We start analysing the simpler case of non-pruned enumeration. We start by recalling to common heuristics used in lattice cryptanalysis.

Heuristic 1 (Gaussian heuristic) *Given a lattice $\Lambda \subset \mathbb{R}^n$ of rank n and a* nice enough *set $S \subset \mathbb{R}^n$, the Gaussian heuristic states that*

$$|\Lambda \cap S| \approx \frac{vol(S)}{covol(\Lambda)}.$$

For cryptanalytic purposes, often S is set to be an n-ball of radius $R \ge 1$. By choosing S to be the n-ball of unit radius, this gives an approximation for the first minimum of Λ:

$$\lambda_1(\Lambda) \approx \frac{\Gamma(1+n/2)^{1/n}}{\sqrt{\pi}}\, covol(\Lambda)^{1/n}. \tag{8}$$

Heuristic 2 (Geometric Series Assumption (GSA) for BKZ [72]) *Let* $B = (b_1, \ldots, b_n)$ *be a BKZ-*β*-reduced basis for a rank-*n *lattice* λ*, and let* $B^* = (b_1^*, \ldots, b_n^*)$ *be the corresponding Gram-Schmidt vectors. Let* $||b_1^*||, \ldots, ||b_n^*||$ *be the* basis profile*. Then the basis profile follows a geometric series*

$$||b_i^*|| \approx \alpha_\beta^{i-1} ||b_1^*||, \text{ where } \alpha_\beta \approx \left((\pi\beta)^{1/\beta} \frac{\beta}{2\pi e}\right)^{-1/(\beta-1)}.$$

Furthermore, by an elementary computation using $covol(\Lambda) = \prod_{i=1}^{n} ||b_i^*||$*, we have*

$$||b_1|| \approx (\alpha_\beta^{-1/2})^{n-1} covol(\Lambda)^{1/n}.$$

Heuristic 3 (BKZ-n-reduced inputs) *Whenever we enumerate a lattice* Λ *of rank* n*, we assume the input basis satisfies Heu. 2, that is that the norms of the Gram-Schmidt vectors follow a geometric series with constant* α_β*.*

We briefly justify making these assumptions in our setting. To simplify our analysis, we will assume that quantum enumeration is being performed as a subroutine to the BKZ lattice reduction algorithm [73,74]. Heu. 1 is commonly assumed, and considered to be accurate already in relatively small dimensions[22], and asymptotically tight [11]. Regarding Heu. 2, while it is known that the output of BKZ does not exactly match the output of the GSA due to fluctuations in the head of the basis profile and a concavity due to the tail being HKZ-reduced, the GSA practically "holds" for most of the basis profile. Regarding Heu. 3, in practice it is known [22,37,50] that the quality of a lattice basis tends to converge to the GSA within the first few tours. Since the runtime of enumeration improves as the basis quality does, we will conservatively assume that in the first few tours the GSA for BKZ-β is achieved, and that from then on all enumeration calls on blocks of rank β satisfy Heu. 3.[10] As we will see, this will imply an asymptotic cost of $2^{\frac{\beta \log \beta}{8} + O(\beta)}$ for enumeration. We note that in practice the $\beta \log \beta / 8$ exponent can indeed be achieved with appropriate local block preprocessing [3].

Armed with Heuristic Assumptions 2 and 3, we can now estimate the asymptotic cost of non-pruned enumeration as an SVP solver in rank n. We start considering an arbitrary enumeration radius R. Let $\text{Ball}_k(\vec{0}, R)$ be the k-ball of

[10] This is an overly optimistic assumption, both because it ignores the HKZ-shape of the basis, that in practice causes a significant slowdown [3], and because we will rely on this analysis for the case of cylinder pruning, where the basis is re-randomised, causing a loss in reduction quality.

radius R centered at the origin. Using the Gaussian heuristic,

$$\begin{aligned} H_k &= \mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}\left[|\mathrm{Ball}_k(\vec{0},R)\cap\pi_{n-k+1}(\Lambda)|\right] \approx \frac{\mathrm{vol}(\mathrm{Ball}_k(\vec{0},R))}{\mathrm{covol}(\pi_{n-k+1}(\Lambda))} \\ &= \mathrm{vol}(\mathrm{Ball}_k(\vec{0},R))\cdot\frac{\prod_{i=1}^{n-k}||b_i^*||}{\mathrm{covol}(\Lambda)} = \mathrm{vol}(\mathrm{Ball}_k(\vec{0},R))\cdot\frac{||b_1^*||^{n-k}\alpha_n^{\frac{(n-k-1)(n-k)}{2}}}{\mathrm{covol}(\Lambda)} \\ &= \frac{R^k\pi^{k/2}}{\Gamma(\frac{k}{2}+1)}\cdot\frac{||b_1^*||^{n-k}\alpha_n^{\frac{(n-k-1)(n-k)}{2}}}{\mathrm{covol}(\Lambda)}. \end{aligned} \tag{9}$$

By letting R be the Gaussian heuristic for the first minimum of the lattice (8), we have

$$\begin{aligned} H_k &\approx \frac{R^k\pi^{k/2}}{\Gamma(\frac{k}{2}+1)}\frac{||b_1^*||^{n-k}\alpha_n^{\frac{(n-k-1)(n-k)}{2}}}{\mathrm{covol}(\Lambda)} \\ &= \frac{\Gamma(\frac{n}{2}+1)^{k/n}}{\pi^{k/2}}\mathrm{covol}(\Lambda)^{k/n}\frac{\pi^{k/2}}{\Gamma(\frac{k}{2}+1)}\frac{||b_1^*||^{n-k}\alpha_n^{\frac{(n-k-1)(n-k)}{2}}}{\mathrm{covol}(\Lambda)} \\ &= \frac{\Gamma(\frac{n}{2}+1)^{k/n}}{\Gamma(\frac{k}{2}+1)}||b_1^*||^{n-k}\alpha_n^{\frac{(n-k-1)(n-k)}{2}}\mathrm{covol}(\Lambda)^{k/n-1} \\ &= \frac{\Gamma(\frac{n}{2}+1)^{k/n}}{\Gamma(\frac{k}{2}+1)}||b_1^*||^{n-k}\alpha_n^{\frac{(n-k-1)(n-k)}{2}}\left(\prod_{i=1}^{n}\alpha_n^{i-1}||b_1^*||\right)^{k/n-1} \\ &= \frac{\Gamma(\frac{n}{2}+1)^{k/n}}{\Gamma(\frac{k}{2}+1)}||b_1^*||^{n-k}\alpha_n^{\frac{(n-k-1)(n-k)}{2}}||b_1^*||^{k-n}\alpha_n^{\frac{(n-1)n}{2}\frac{k-n}{n}} \\ &= \frac{\Gamma(\frac{n}{2}+1)^{k/n}}{\Gamma(\frac{k}{2}+1)}\alpha_n^{\frac{k(k-n)}{2}}. \end{aligned}$$

Using Stirling's approximation for the Γ function on $\mathbb{R}^+$,

$$\Gamma(x+1) = \sqrt{2\pi x}\left(\frac{x}{e}\right)^x\left(1+\frac{1}{12x}+O\left(\frac{1}{x^2}\right)\right),$$

we can further simplify the result as

$$H_k \approx \frac{\Gamma(\frac{n}{2}+1)^{k/n}}{\Gamma(\frac{k}{2}+1)}\cdot\alpha_n^{\frac{k(k-n)}{2}} \approx \frac{(\pi n)^{\frac{k}{2n}}\left(\frac{n}{2e}\right)^{\frac{k}{2}}}{\sqrt{\pi k}\left(\frac{k}{2e}\right)^{\frac{k}{2}}}\cdot\alpha_n^{\frac{k(k-n)}{2}} \approx (\pi n)^{\frac{1}{2}\left(\frac{k}{n}-1\right)}\left(\frac{n}{k}\right)^{\frac{k}{2}}\cdot\alpha_n^{\frac{k(k-n)}{2}} \tag{10}$$

In [32], it is observed that most of the nodes are located around level $k \approx n/2$. There, the approximation becomes $N \approx H_{n/2} \approx (\pi n)^{-\frac{1}{4}}2^{\frac{n}{4}}\cdot\alpha_n^{-\frac{n^2}{8}}$. Using the "rule of thumb" approximation $\alpha_n^{-\frac{1}{2}} \approx n^{\frac{1}{2n}}$, the leading term in $H_{n/2}$ becomes $n^{\frac{1}{2n}\frac{n^2}{4}} = 2^{\frac{n}{8}\log n}$. Using Sagemath's [77] `find_fit` function over the approximation derived from (10) for $N = \frac{1}{2}\sum_{k=1}^{n}H_k$ (with α_n as in Heu. 2) evaluated it

on $\{100, 200, \ldots, 1000\}$, we find

$$\log N \approx 0.12463\, n \log n - 0.25483\, n + 0.35621\, \log n - 0.26238.$$

A.2 Computing H_k Lower Bounds when Cylinder Pruning is Used

The analysis for cylinder pruning is slightly more complex, since cylinder pruning really indicates a potentially vast family of pruning strategies, depending on how the pruning radii $0 < R_1 \leq \cdots \leq R_n = R$ are chosen. Gama *et al.* [32] propose using either linear pruning, where $R_i^2 = (i/n) \cdot R^2$, for $i = 1, \ldots, n$, piece-wise linear functions, or numerically optimised ones. While for some choices, such as linear pruning, it may be possible to find closed formulae for the volume of the cylinder intersections $C_{R_1,\ldots,R_k}$, this is not necessarily possible in general.

In [12], Aono *et al.* propose a technique for computing lower bounds to the complexity of cylinder pruning in general, for which they are able to give a closed formula [12, Eq. 16], in terms of the inverse regularized incomplete beta function and of the values for H_k in the case of no pruning. Aono *et al.* check the tightness of their results, concluding that their results should be reasonably tight. Given the ease of computing the lower bounds in [12] and their relative tightness, we use them to estimate a lower bound to H_k numerically as part of our cost estimations.

B Estimating the Size of Lattice Enumeration Subtrees

In this section we elaborate on the number of descendants of a node $|W_{k,h}(g)|$, in general for $h \geq 1$ and in particular for the number of children $C(g) = |W_{k,1}(g)|$.

B.1 Theoretical Analysis

We start by providing a proof of Lem. 1.

Proof (of Lem. 1). Let $\mathbb{Z}_+$ be the set of positive integers, and $\mathbb{Q}_+$ be the set of positive rationals. We start by observing that the $W_{k,h}(g)$ partition the set Z_{k+h}, since every element in the latter descends from a unique element $g \in Z_k$. By the definition of expectation,

$$\mathop{\mathbb{E}}_{\substack{g \sim U(Z_k) \\ \text{rand. } \mathcal{T}}} [|W_{k,h}(g)|] = \mathop{\mathbb{E}}_{\substack{\text{random} \\ \text{tree } \mathcal{T}}} \left[\sum_{g \in Z_k} \frac{|W_{k,h}(g)|}{|Z_k|} \right] = \mathop{\mathbb{E}}_{\substack{\text{random} \\ \text{tree } \mathcal{T}}} \left[\frac{|Z_{k+h}|}{|Z_k|} \right].$$

We then note that as random variables, the supports $\mathrm{supp}(Z_k)$, $\mathrm{supp}(Z_{k+h}) \subset \mathbb{Z}_+$, while $\mathrm{supp}(Z_{k+h}/Z_k) \subset \mathbb{Q}_+$. We then proceed to compute

$$\begin{aligned}
\mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}\left[\frac{|Z_{k+h}|}{|Z_k|}\right] &= \sum_{r\in\mathbb{Q}_+} r \Pr\left[\frac{|Z_{k+h}|}{|Z_k|} = r\right] \\
&= \sum_{r\in\mathbb{Q}_+} r \sum_{y\in\mathbb{Z}_+} \Pr\left[\frac{|Z_{k+h}|}{|Z_k|} = r \mid |Z_k| = y\right] \Pr\left[|Z_k| = y\right], \\
&= \sum_{r\in\mathbb{Q}_+} \sum_{y\in\mathbb{Z}_+} \frac{1}{y} ry \Pr\left[|Z_{k+h}| = ry\right] \Pr\left[|Z_k| = y\right], \\
&= \sum_{y\in\mathbb{Z}_+} \frac{1}{y} \Pr\left[|Z_k| = y\right] \sum_{r\in\mathbb{Q}_+} ry \Pr\left[|Z_{k+h}| = ry\right],
\end{aligned}$$

where the summations commute due to the summands being non-negative [68, Cor. to Thm. 1.27]. We note that given any $y \in \mathbb{Z}_+$, we have $y \cdot \mathbb{Q}_+ = \mathbb{Q}_+$, meaning that we can simplify

$$\sum_{r\in\mathbb{Q}_+} ry \Pr\left[|Z_{k+h}| = ry\right] = \mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}\left[|Z_{k+h}|\right] = H_{k+h}.$$

We also observe that

$$\sum_{y\in\mathbb{Z}_+} \frac{1}{y} \Pr\left[|Z_k| = y\right] = \sum_{y\in\mathbb{Z}_+} \frac{1}{y} \Pr\left[\frac{1}{|Z_k|} = \frac{1}{y}\right] = \mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}\left[\frac{1}{|Z_k|}\right] \geq \frac{1}{\mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}\left[|Z_k|\right]} = \frac{1}{H_k},$$

where the last inequality is Jensen's inequality applied to $f(|Z_k|) = 1/|Z_k|$. Putting these two results together, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } \mathcal{T}}}\left[\frac{|Z_{k+h}|}{|Z_k|}\right] &= \sum_{y\in\mathbb{Z}_+} \frac{1}{y} \Pr\left[|Z_k| = y\right] \sum_{r\in\mathbb{Q}_+} ry \Pr\left[|Z_{k+h}| = ry\right] \\
&= \sum_{y\in\mathbb{Z}_+} \frac{1}{y} \Pr\left[|Z_k| = y\right] H_{k+h} \geq \frac{H_{k+h}}{H_k}.
\end{aligned}$$

□

We then obtain an upper bound to the number of children of a node.

Lemma 2 (Upper bound of $C(g)$). *Let $R_1 \leq \cdots \leq R_n$ be the enumeration radii in pruned enumeration. Let B be a basis of the lattice Λ of rank n. Let g_{j+1} be a guess for $\pi_{j+1}(v)$ on level $k = n-(j+1)+1$ of the enumeration tree . The number of children of g_{j+1} on the tree is at most approximately $\lfloor x \cdot R_{k+1}/||b_j^*|| \rceil = \lfloor x \cdot R_{k+1}/||b_{n-k}^*|| \rceil$, were $x = 1$ if $k = 0$ and 2 otherwise.*

Proof. Recall Eq. (1):

$$\left|c_j + \sum_{i>j} \mu_{i,j}\, c_i\right| \leq \frac{1}{||b_j^*||}\sqrt{R_{k+1}^2 - ||\pi_{j+1}(v)||^2}$$
$$= \frac{1}{||b_j^*||}\sqrt{R_{k+1}^2 - \sum_{r=j+1}^{n}\left(c_r + \sum_{i=r+1}^{n} \mu_{i,r} c_i\right)^2 ||b_r^*||^2}.$$

Guessing a value g_j for $\pi_j(v)$ consists of picking $c_j \in \mathbb{Z}$ satisfying Eq. (1) given fixed values for c_i for $i > j$ and for Gram-Schmidt vectors and coefficients. Letting RHS be the right-hand side of Eq. (1), $c_j \in [-\text{RHS} - \sum_{i>j} \mu_{i,j}\, c_i, \text{RHS} - \sum_{i>j} \mu_{i,j}\, c_i] \cap S$ where $S = \mathbb{Z}$ if $k > 0$ and $S = \mathbb{Z}_{\geq 0}$ if $k = 0$. The size of such intersection, and hence the number of valid guesses for c_j, is approximately $\lfloor x \cdot \text{RHS} \rceil$ for $x = 2$ if $k > 0$ and $x = 1$ otherwise. Finally, we note that $\text{RHS} \leq R_{k+1}/||b_j^*||$, proving the result. □

Expected value of $\mathbb{E}[|W_{k,h}(g)|]$. As part of our experiments on non-pruned enumeration for verifying Lems. 1 and 2, we observed that Lem. 1 appears tight for $h \approx 1$. We present an attempt at a proof of this heuristic, pointing out the argument that we are missing for completing it.

Heuristic 4 *The lower bound of Lem. 1 is tight for $h \approx 1$, i.e.*

$$\mathop{\mathbb{E}}_{g \sim U(Z_k)}[|W_{k,h}(g)|] = \mathop{\mathbb{E}}_{\substack{random \\ tree\ T}}\left[\frac{|Z_{k+i}|}{|Z_k|}\right] \approx \frac{H_{k+h}}{H_k} \quad \textit{for } h \approx 1.$$

Proof. (of Heu. 4; incomplete) The first equality follows from the proof of Lem. 1. The approximation follows from taking the expectation of the Taylor series of $\frac{x}{y}$ evaluated at $(x, y) = (\mathbb{E}[|Z_{k+h}|], \mathbb{E}[|Z_k|])$ [Eq. 3.87-88][27].

The error term in the approximation is quadratic in $(|Z_{k+h}| - \mathbb{E}[|Z_{k+h}|])$ and $(|Z_k| - \mathbb{E}[|Z_k|])$. An argument that these estimates are tight would complete the proof, however they would also imply tightness for general h, and not just for $h \approx 1$. ↯

B.2 Experimental Evidence

In the following, we provide experimental evidence for the accuracy of Heu. 4 when $h = 1$, as well as for the correctness of Lems. 1 and 2, in the setting of lattice enumeration *without pruning*. To verify our heuristic arguments, we modified a copy of `fplll` version 5.4.2 [76] to record the average number of children per node during enumeration, as well as the average size of subtrees $W_{k,h}(g)$ rooted at some given level k, and of height h. We run experiments on lattices build using the primal attack embedding by Bai and Galbraith [15][11]

[11] Positioning the q-vectors at the beginning of the basis.

on learning with errors (LWE) instances estimated to be solved using BKZ with block size $\beta = 60$. Given the first minimum λ_1 of the blocks being enumerated, the enumeration radius is set to $R = 1.1 \cdot \lambda_1$. We list the parameters in Tab. 7.

Since Lem. 1 requires computing H_k, we run our experiments on trees generated by enumeration without pruning, to avoid the issue of having to compute the volumes of the cylinder intersections used in pruned enumeration as part of the Gaussian heuristic. While this means that our predictions for the lower bound are easier to compute, it also means that we run experiments only up to block size $\beta = 40$.

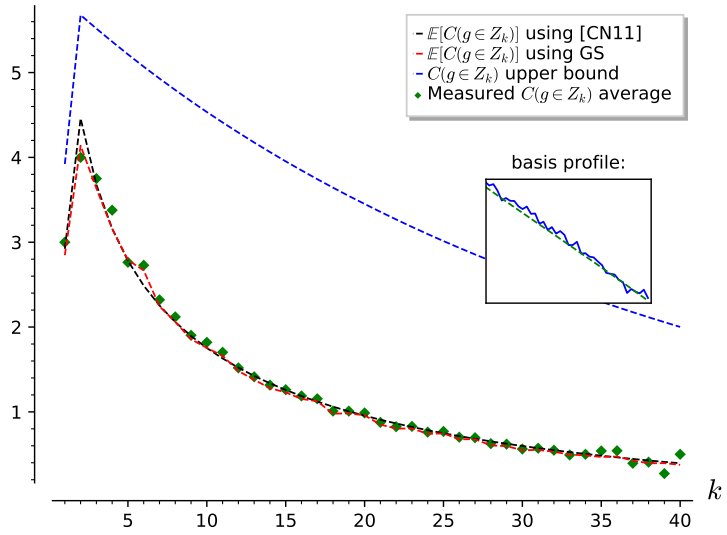
Table 7: LWE parameters used in lattice reduction experiments. The standard deviation σ was used for instantiating discrete gaussian samplers for errorr and secret vector coefficients.

secret dimension n	modulo q	std. dev. σ	samples m
72	97	1.0	87

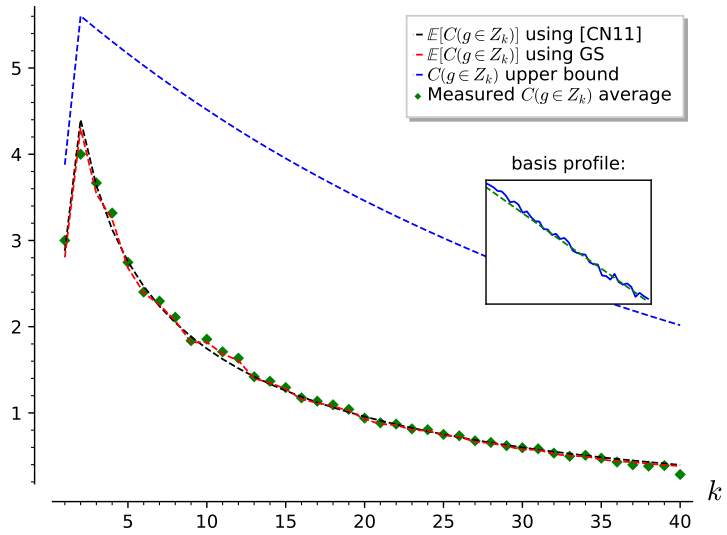
Remark 4. Our theoretical treatment of lattices happens in a generous model where lattice bases achieve the GSA corresponding to BKZ-β reduced-ness at all basis indices. Experimentally, this is not easily achievable. To account for this we take a few precautions:

- We inspect experiment results starting at 10 completed tours of BKZ-β, in order to give some time for the basis profile to approach the GSA.
- We inspect experiments happening on blocks located approximately half-way through the basis, in order to protect them from GSA deviations:
  - Towards the initial bases vectors due to the presence of several q-vectors;
  - Towards the final β bases vectors due to these being HKZ reduced, causing a clear deviation from the GSA.
- We select the highest possible block sizes `fplll` would successfully execute within a reasonable time, to enter the Gaussian heuristic "regime" as much as possible; unfortunately due to checking heuristics for enumeration without pruning this means attempting relatively small block sizes.
- As an extra show of caution we over-impose the a plot showing the basis profile on the block being enumerated for every experiment, and the [22] simulation of the corresponding block.

Results on $C(g) = |W_{k,1}(g)|$. We observe close agreement between our heuristics and the measurements at block size $\beta = 40$. In Fig. 5, we provide two example plots of a characteristic measurement.



(a) $n = 72$ (cf. Tab. 7), block size $\beta = 40$, 10^{th} tour, block $\pi_{80}(b_{80}, \ldots, b_{80+\beta-1})$.



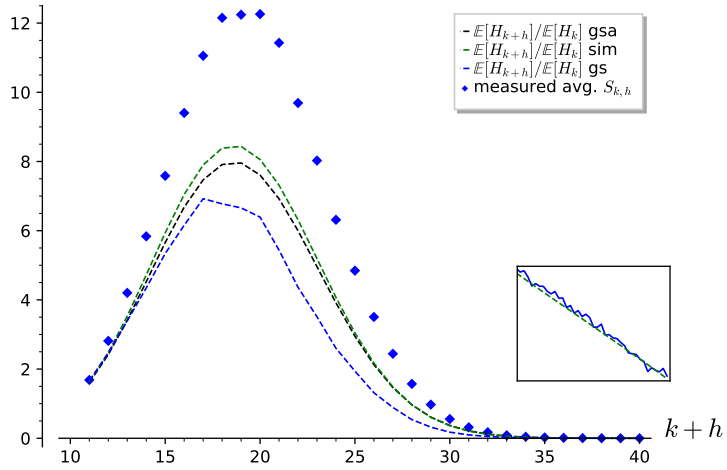
(b) $n = 72$ (cf. Tab. 7), block size $\beta = 40$, 15^{th} tour, block $\pi_{70}(b_{70}, \ldots, b_{70+\beta-1})$.

Figure 5: Plots displaying our predictions and measurements for the number of children $C(g)$ of a node $g \in Z_k$ as a function of k. The green dots are the averages we measure during the call to enumeration. The blue dashed line is our upper bound for $C(g)$ given by Lem. 2. Red and black dashed lines are $\mathbb{E}[C(g)]$ based on Heu. 4 using the measured $||b_k^*||$ and a prediction based on the [22] simulator, respectively. In the sub-plot, the green dashed line is the BKZ simulator's [22] prediction for the basis profile $\log ||b_k^*||$, the blue line is our measurement.

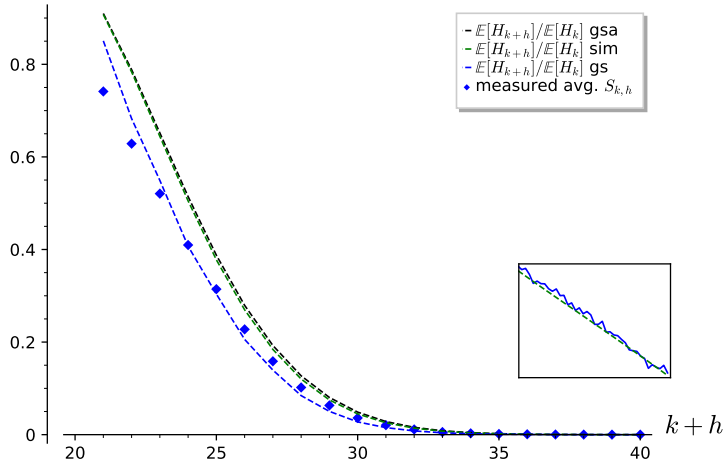
Results on $|W_{k,h}(g)|$, when $h > 1$. In Figs. 6 to 8, we provide example plots of characteristic measurements. In order to imitate the combined classical-quantum methodology from § 3, we pick subtrees reaching towards the lowest level on the tree. This means fixing a height $h \in \{20, 30\}$, and picking $k = \beta - h$ when measuring statistics.

We compute Lem. 1 three times per experiment: using the measured basis profile, using a simulated basis profile following [22], and using the geometric series assumption, in order to observe how much the predictions can differ. We observe that our experiments deviate from the expected results more than in the case of $C(g)$.

Given the caveats listed in Rmk. 4, we believe that the measured data should be compared to the prediction for the lower bound obtained using the measured basis profile. Out of the cases presented, those where $k = 10, h = 30$ satisfy the lower bound in Lem. 1 using the measured profile. This is not always the case for the $k = h = 20$ case, where some deviation can be seen, although the general shape seems to be captured by the predictions. We don not believe that this should be considered contradiction of Lem. 1, but rather an inherent limitation of computing Lem. 1 using the Gaussian heuristic while having to run experiments in the low-dimensional regime where the heuristics may not be fully precise.

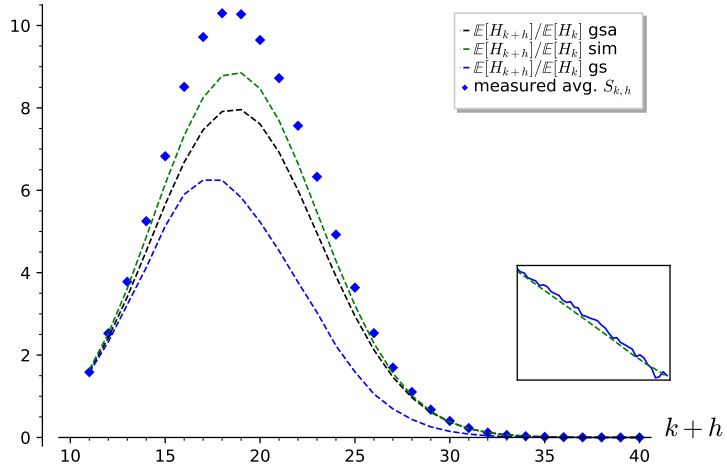


(a) $n = 72$ (cf. Tab. 7), block size $\beta = 40$, 10^{th} tour, block $\pi_{80}(b_{80}, \ldots, b_{80+\beta-1})$, $k = 10$.

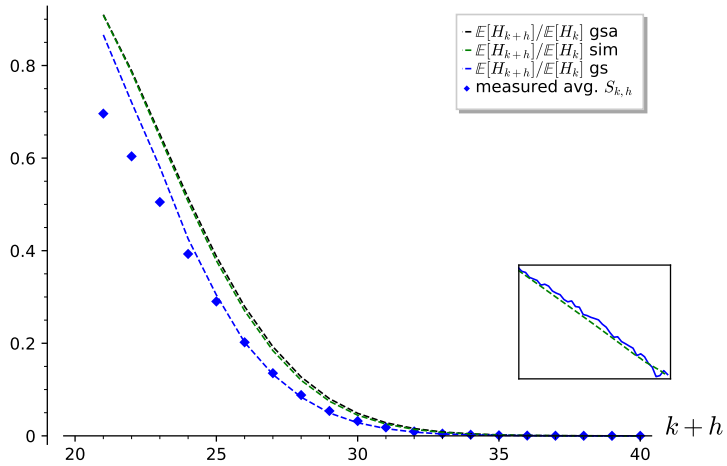


(b) $n = 72$ (cf. Tab. 7), block size $\beta = 40$, 10^{th} tour, block $\pi_{80}(b_{80}, \ldots, b_{80+\beta-1})$, $k = 20$.

Figure 6: Plots displaying our predictions and measurements for the number of descendants $|W_{k,h}(g)|$ on level $k+h$ of a node $g \in Z_k$ as a function of $k+h$. The blue dots are the averages we measure during the call to enumeration. The blue dashed line is our lower bound for $C(g)$ given by Lem. 1, computed using the measured $||b_k^*||$ to evaluate H_k and H_{k+h} using the Gaussian heuristic. Green and black dashed lines are the same lower bound computed using the BKZ simulator's [22] simulator and the geometric series assumption, respectively. In the sub-plot, the green dashed line is the BKZ simulator's [22] prediction for the basis profile $\log ||b_k^*||$, the blue line is our measurement.

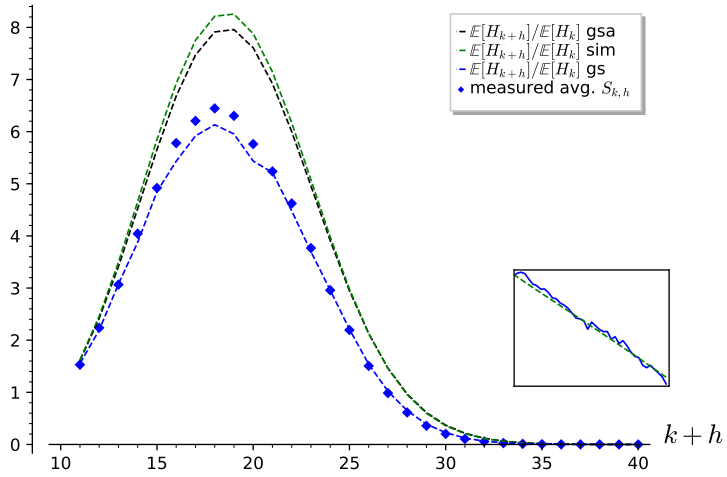


(a) $n = 72$ (cf. Tab. 7), block size $\beta = 40$, 10^{th} tour, block $\pi_{90}(b_{90}, \ldots, b_{90+\beta-1})$, $k = 10$.

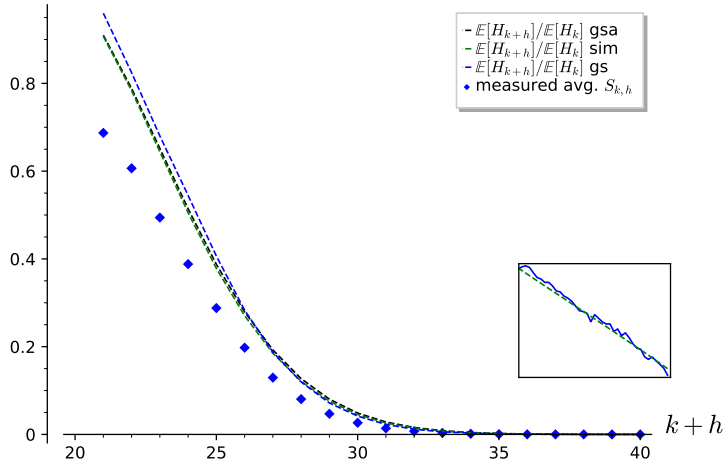


(b) $n = 72$ (cf. Tab. 7), block size $\beta = 40$, 10^{th} tour, block $\pi_{90}(b_{90}, \ldots, b_{90+\beta-1})$, $k = 20$.

Figure 7: Continuation of Fig. 6.



(a) $n = 72$ (cf. Tab. 7), block size $\beta = 40$, 15^{th} tour, block $\pi_{80}(b_{80}, \ldots, b_{80+\beta-1})$, $k = 10$.



(b) $n = 72$ (cf. Tab. 7), block size $\beta = 40$, 15^{th} tour, block $\pi_{80}(b_{80}, \ldots, b_{80+\beta-1})$, $k = 20$.

Figure 8: Continuation of Fig. 6.

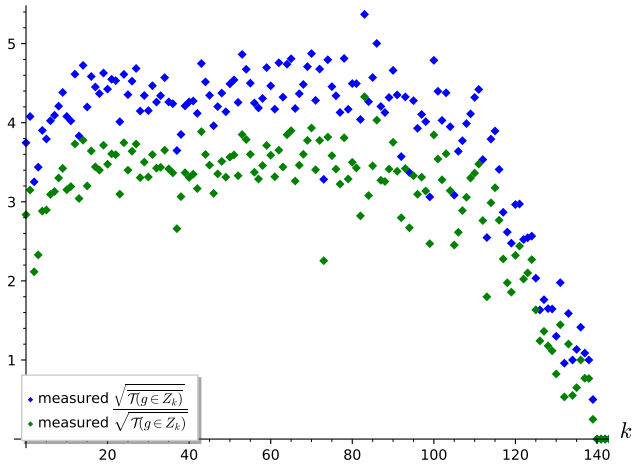
C Experiments on the Multiplicative Jensen Gap

In § 5, we presented approximate lower bounds on the cost of combined classical-quantum enumeration, identifying values of the multiplicative Jensen gap for which the attacks would achieve certain threshold complexities. We displayed these in Tab. 6, with more present in Tab. 9 in App. F.

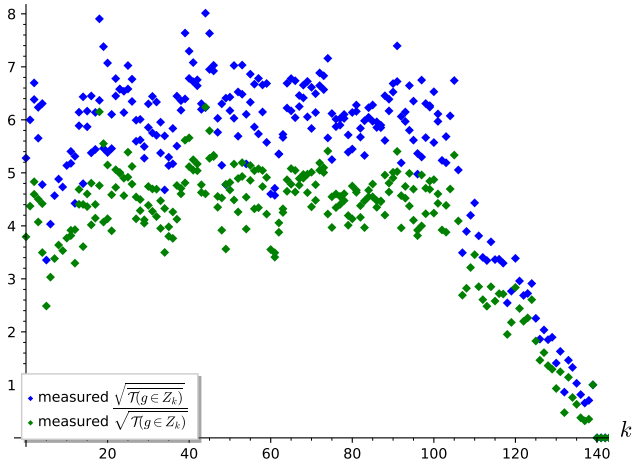
In this appendix, we present some experiments run on pruned enumeration trees using `fplll` version 5.4.2 [76], on the same lattices used in App. B. We measured the multiplicative Jensen gap for subtrees of height h rooted on level k for varying k, for every enumeration tree explored during BKZ reduction. Unlike in App. B, in this case we do not need to predict H_k, meaning that we can use pruned enumeration as set by default in `fplll`. This means that we can run experiments for block sizes $\beta > 40$.

We present our measurements in Figs. 9 and 10. We observe that the Jensen gap in the experiments we have attempted seems to be around $2^z \leq 2$. While our experiments are in low dimension and therefore cannot be extrapolated into cryptanalytic claims, they also suggest that the possibility of enumeration tree subtrees achieving small multiplicative Jensen gaps cannot be immediately disregarded.

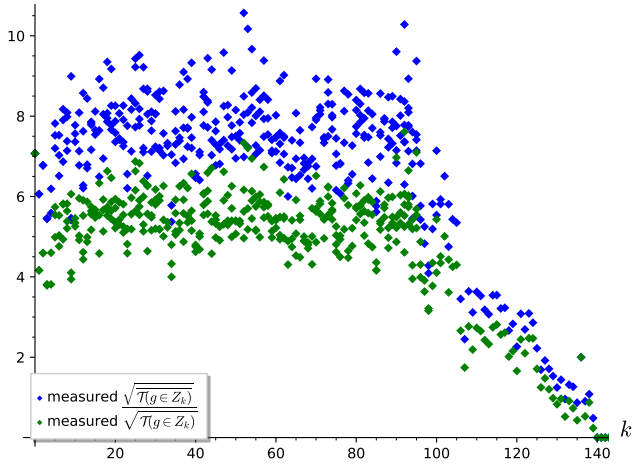
Remark 5. As the block size increases, more rerandomisation is performed by `fplll`'s default pruning strategy, consequently resulting in a denser dataset for $\beta = 70$ than for $\beta = 50$.



(a) $n = 72$ (cf. Tab. 7), block size $\beta = 50$, 10^{th} tour, $h = 20$.

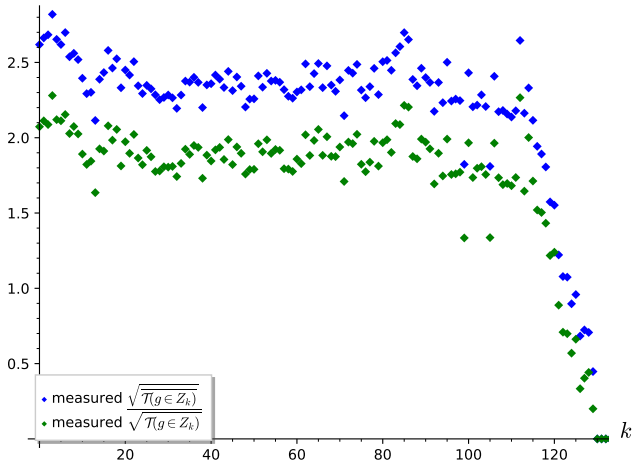


(b) $n = 72$ (cf. Tab. 7), block size $\beta = 60$, 10^{th} tour, $h = 20$.

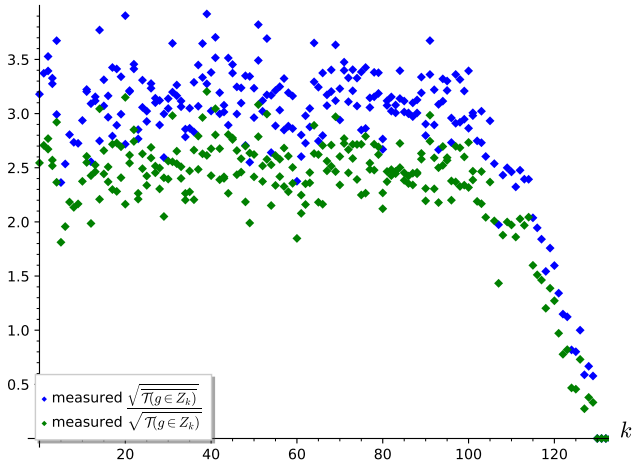


(c) $n = 72$ (cf. Tab. 7), block size $\beta = 70$, 10^{th} tour, $h = 20$.

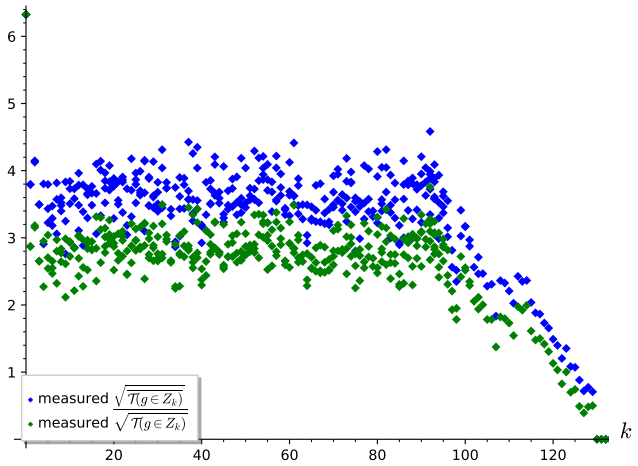
Figure 9: Measurement on the Jensen gap of subtrees of height $h = 20$ in pruned enumeration experiments. The height h is reduced in the last few blocks of the basis of dimension smaller than h. Blue dots are the squared root of the average, green dots are the average of the squared root.



(a) $n = 72$ (cf. Tab. 7), block size $\beta = 50$, 10^{th} tour, $h = 30$.



(b) $n = 72$ (cf. Tab. 7), block size $\beta = 60$, 10^{th} tour, $h = 30$.



(c) $n = 72$ (cf. Tab. 7), block size $\beta = 70$, 10^{th} tour, $h = 30$.

Figure 10: Measurement on the Jensen gap of subtrees of height $h = 30$ in pruned enumeration experiments. The height h is reduced in the last few blocks of the basis of dimension smaller than h. Blue dots are the squared root of the average, green dots are the average of the squared root.

D Instantiation of $\mathcal{W}$

In this appendix, we add a few more details to the analysis done in § 4.

D.1 Components of $\mathcal{W}$

Given a tree $\mathcal{T}$ of height h, Montanaro [55] defines the operator $\mathcal{W}$ using two operators R_A and R_B, the first acting on all nodes with even distance from the root, and the second on all nodes with odd distance. The implementations of the two operators are nearly identical, and as such we will only capture the original description of R_A [55, Alg. 3] by decomposing it into the following operators:

$U_{\mathbf{Setup}}$: quantum operator that prepares the quantum state by advancing the variable assignment (*i.e.*, level k of the tree), and ensures that the operator acts on the *correct* set of nodes (*i.e.*, even or odd levels for $U_{\mathrm{Setup}(R_A)}$ or $U_{\mathrm{Setup}(R_A)}$, respectively).

$U_{\alpha,S}$: quantum operator that generates a superposition of children of a node, performing the map $|0\rangle \mapsto |\phi_{\alpha,S}\rangle$, where S is the set of all children of a node.

$U_{\mathbf{P}}$: quantum operator that computes the norm of the projected lattice point being inspected, and compares the length of the projection with the pruning bound R_i. The predicate is executed to identify the children of a node.

U_0: reflection through $|0\rangle$. Together with the operator $U_{\alpha,S}$ it performs the the diffusion operation $I - 2\,|\phi_{\alpha,S}\rangle\,\langle\phi_{\alpha,S}|$.

$U_{\mathbf{Uncompute}}$: quantum operator to uncompute the ancillary states and the inversion of the setup step U_{Setup}.

D.2 Quantum Arithmetic

Smallest known arithmetic circuits. The quantum arithmetic literature contains many design proposals for integer and floating point adders and multipliers. Generally, most algorithms are either "ports" of classical designs [26,35,36,59,57], or they rely on the quantum Fourier transform (QFT) to evaluate these operations [69,45]. As not all papers work using the same metrics or even quantum computing architectures, direct comparisons and trade-off evaluations can be difficult. For example, Pham and Svore [61] claim additions in constant depth and multiplications in logarithmic depth, however this seems to require a specifically designed quantum architecture. For a rule-of-thumb estimation of our attack costs, we opt to chose potentially more common asymptotics for adders and multipliers achieving the smallest T counts and depths in our literature review (other than [61]). We report these in Tab. 1, and ignore constants and lower order terms hidden by the $\mathcal{O}$. Whenever numbers of different bit lengths are multiplied, we conservatively assume both to have the smaller length, since we have not found sources describing quantum circuits for unbalanced multiplication. For the "$x \leq y$" comparison operator, we use a circuit with the same asymptotic size of an adder [26].

Floating point numbers. We also define a value ξ that equals the amount of floating point precision required to store the coefficient of the basis vectors b_i. The literature on this topic either proposes algorithms for estimating the required precision [65,25], appearing this to be about $\Theta(n)$, or uses double precision [38]. We notice that while the `fplll` library [76] includes support for arbitrary precision floating point numbers, often experiments use double- or quadruple-precision floating point numbers. As a conservative choice, we observe that any setting where quantum-enumeration would be advantageous would likely result in requiring more than double-precision, since otherwise cheap classical implementations are available, and therefore consider $\xi = 53$ to be a lower bound on the required precision.

D.3 Arithmetic Cost

Table 8: Assumed cost of arithmetic operations in $U_{\mathrm{P}}^{\min}$. Each operations has input numbers of bit length x_i and outputs numbers of size x_{i+1}

Operation	Input bit lengths	T-Depth	GCost
$(c_i, (\vec{b}_i)_j) \mapsto c_i(\vec{b}_i)_j$	$x_0 = \min(\mathfrak{b}, \xi)$	$\log^2(x_0)$	$h^2 \cdot x_0 \log(x_0) \log(\log(x_0))$
$(c_i(\vec{b}_i)_j)_i \mapsto \sum_i c_i(\vec{b}_i)_j$	$x_1 = \mathfrak{b} + \xi$	$\log h \cdot \log(x_1)$	$h^2 \cdot x_1$
$\sum_i c_i(\vec{b}_i)_j \mapsto (\sum_i c_i(\vec{b}_i)_j)^2$	$x_2 = \mathfrak{b} + \xi + \log h$	$\log^2(x_2)$	$h \cdot x_2 \log(x_2) \log(\log(x_2))$
$((\sum_i c_i(\vec{b}_i)_j)^2)_j \mapsto \|\|\sum_i c_i\vec{b}_i\|\|^2$	$x_3 = 2(\mathfrak{b} + \xi + \log h)$	$\log h \cdot \log(x_3)$	$h \cdot x_3$
$\|\|\sum_i c_i\vec{b}_i\|\|^2 \leq R^2 - \|\|g\|\|^2$	$x_4 = 2(\mathfrak{b} + \xi + \log h) + \log h$	$\log(x_4)$	x_4

Depth and cost estimation of $\mathcal{W}$. Given a tree $\mathcal{T}$ of height h, Montanaro [55] defines the operator $\mathcal{W}$ using two operators R_A and R_B, the first acting on all nodes with even distance from the root, and the second on all nodes with odd distance. The implementations of the two operators are nearly identical, and as such we will only capture the original description of R_A [55, Alg. 3] by decomposing it into the following operators:

In our setting, quantum enumeration is being performed on a tree $\mathcal{T}(g \in Z_k)$ of height $h = n - k$, where n is the dimension of the full lattice Λ being enumerated. This process would require performing arithmetic using the projected lattice basis vectors $(\pi_{n-k+1}(b_{k+1}), \ldots, \pi_{n-k+1}(b_n))$ of Λ. In an attempt to unburden notation, in this paragraph we temporarily relable these as $(b_1, \ldots, b_h)$, and consider them to be h-dimensional by applying an appropriate rotation. Our estimate for the cost of each component of $W^{\min}$ is then:

$U_{\mathbf{Setup}}$: sets up the states, we assume $\mathrm{GCost}(U_{\mathrm{Setup}}) = \text{T-}\mathrm{Depth}(U_{\mathrm{Setup}}) = 0$.

$U_{\alpha,S}^{\mathbf{min}}$: generates a superposition of the children of a node. At a bare minimum, this requires a uniform superposition $\sum_{i=0}^{q-1} |i\rangle$ which is computed

by a single (parallel) layer of Hadamard gates. Since we are only accounting for T, we assume $\text{T-DEPTH}(U^{\min}_{\alpha,S}) = \text{T-DEPTH}(U^{\min}_{\mathrm{P}})$ and $\text{GCOST}(U^{\min}_{\alpha,S}) = \text{GCOST}(U^{\min}_{\mathrm{P}})$, as Hadamard gates don't contribute T gates.

$U^{\min}_{\mathbf{P}}$: evaluates the predicate P which identifies projected vectors $v \in \pi_{n-\ell+1}(\Lambda)$ such that $||v|| \leq R_\ell$ and $\pi_{n-k+1}(v) = g$, for all levels $k < \ell \leq n$. In order to lower-bound the cost of this operation, we only consider the case of evaluating this inequality at $\ell = n$, where the condition becomes checking whether $||\sum_{i\leq h} c_i \vec{b}_i||^2 \leq R^2 - ||g||^2$.[12] The cost of the operation needs to account for at least the cost of the following operations (summarised in Tab. 8):

1. Parallel multiplication of h^2 pairs $(c_i, (\vec{b}_i)_j) \mapsto c_i(\vec{b}_i)_j$, of $\mathfrak{b}$- and ξ-bit length, outputting numbers of bit length $\mathfrak{b} + \xi$.
2. Addition of coefficients $(c_1(\vec{b}_1)_j, \ldots, c_h(\vec{b}_h)_j) \mapsto \sum_i c_i(\vec{b}_i)_j$ for $j \in [h]$. These additions can be run in parallel over j. For a fixed j, the corresponding sum is run by adding terms in pairs, forming a binary tree of sums. Each $\sum_i c_i(\vec{b}_i)_j$ output is $\mathfrak{b} + \xi + \log h$ bits long.
3. Squaring the $\sum_i c_i(\vec{b}_i)_j$ sums in parallel (output bit length $2(\mathfrak{b}+\xi+\log h)$) and adding them in a binary-tree fashion to obtain $||\sum_{i\leq h} c_i\vec{b}_i||^2 = \sum_j(\sum_i c_i(\vec{b}_i)_j)^2$ of bit length $2(\mathfrak{b} + \xi + \log h) + \log h$.
4. The last operation is the comparison with the (adjusted) pruning bound $R^2 - ||g||^2$.

We depict the implementation of the minimal $U^{\min}_{\mathrm{P}}$ implementation in Fig. 11.

U_0: the quantum operator computing $2\,|0\rangle\langle 0| - \mathrm{Id}$ which requires mult-controlled-Z gates, we estimate requiring at least one T gate.

$U_{\mathbf{Uncompute}}$: we conservatively assume that uncomputation does not require T gates, as in the case of measurement-based uncomputation of AND gates in [39], and assume $\text{T-DEPTH}(U_{\text{Uncompute}}) = \text{GCOST}(U_{\text{Uncompute}}) = 0$.

[12] In classical implementations, this computation benefits from extensive caching of Gram-Schmidt orthogonalisation operations and results [76]. Asymptotically, the number of individual arithmetic operations is the same as if computing directly from the basis $(b_1, \ldots, b_h)$.

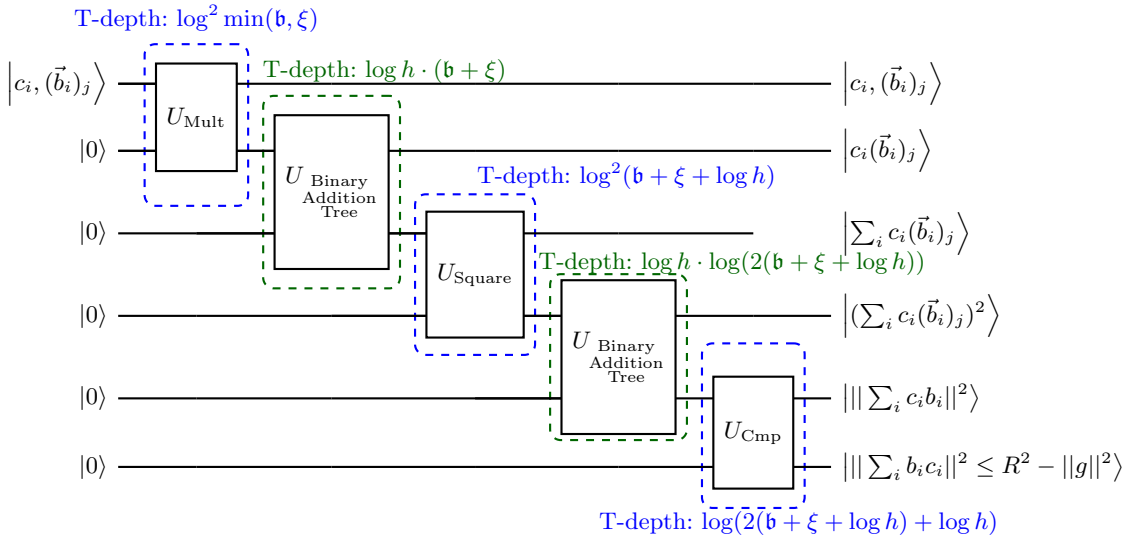


Figure 11: Minimal quantum circuit of $U_{\text{P}}^{\min}$.

E Results from Kyber Parameters

E.1 Figures for the Query-based Result

In this section we present the figures from our estimations of the cost of quantum enumeration that we have not reported in § 5. For all cost estimations the quantum operator GCost($\mathcal{W}$) and TDepth($\mathcal{W}$) are estimated as in § 4.1 and with $DF(\mathcal{W})$, $QD(\mathcal{W})$, $WQ(\mathcal{T}, W)$ as in § 3,

- Figure 12 is the cost estimation for Kyber-512 with T-Depth(QPE($\mathcal{W}$)) $\leq 2^{64}$
- Figure 13 shows the cost estimation without any MaxDepth constraint on T-Depth(QPE($\mathcal{W}$))
- Figure 14 shows the cost estimation for Kyber-768 and Kyber-1024 with T-Depth(QPE($\mathcal{W}$)) $\leq \{2^{40}, 2^{64}, 2^{96}\}$ respectively.

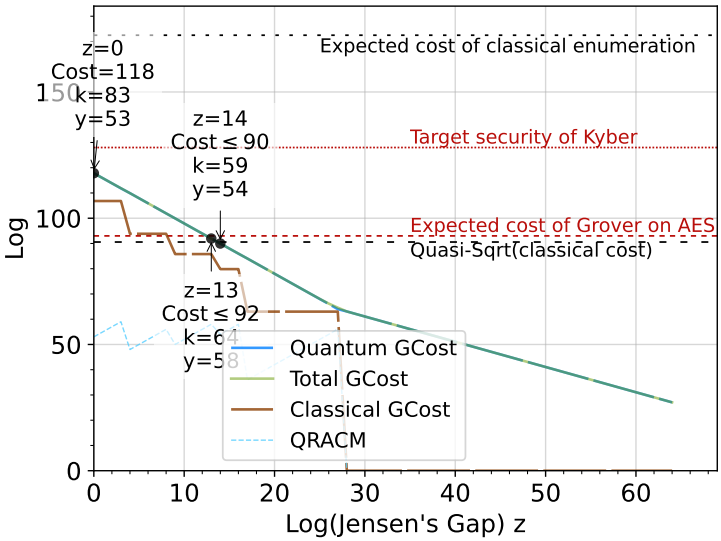
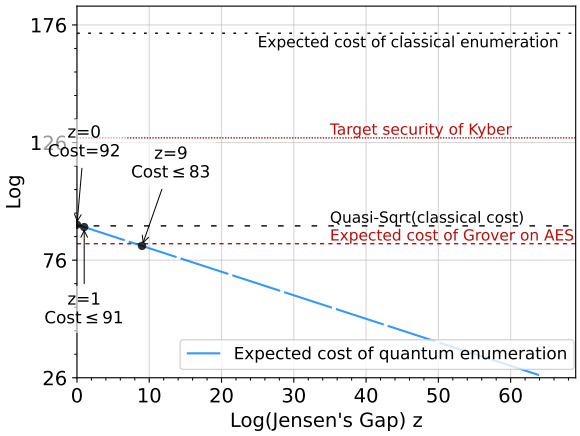
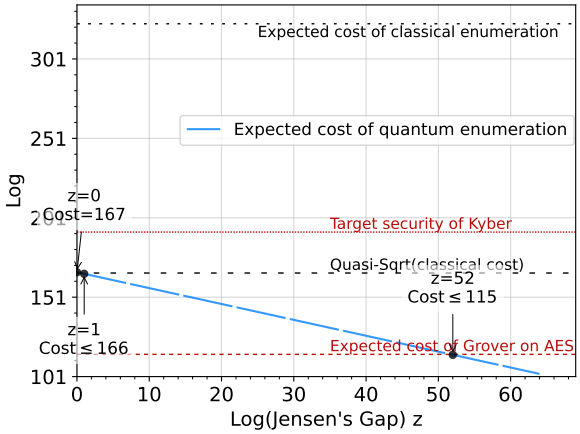


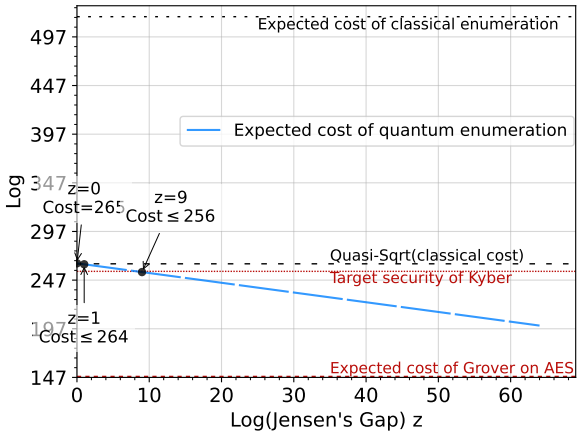
Figure 12: Cost estimation for Kyber with MaxDepth $= 2^{64}$ with the instantiation for operator $\mathcal{W}$ as in § 4.1 and with DF $= 1$, QD $= 1$, $b = 1$ (see § 3).



(a) Cost estimation for Kyber-512 without MaxDepth restrictions.

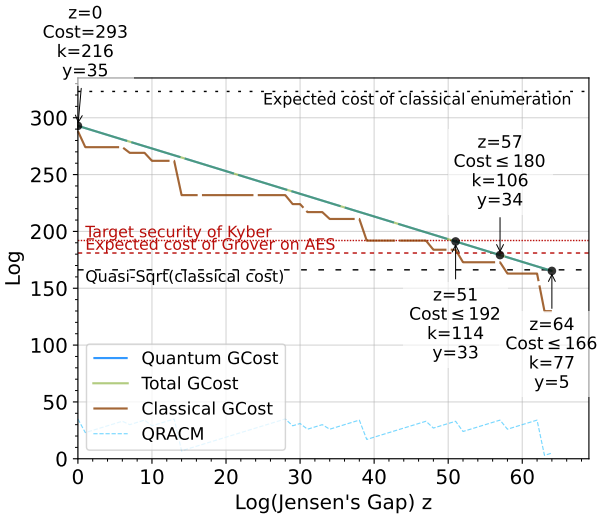


(b) Cost estimation for Kyber-768 without MaxDepth restrictions.

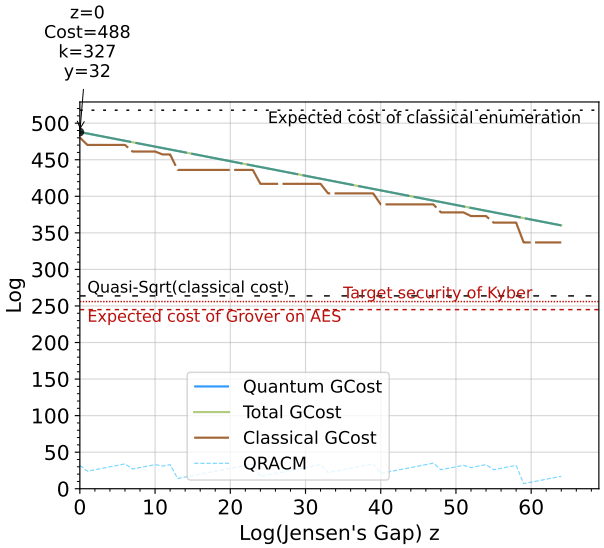


(c) Cost estimation for Kyber-1024 without MaxDepth restrictions.

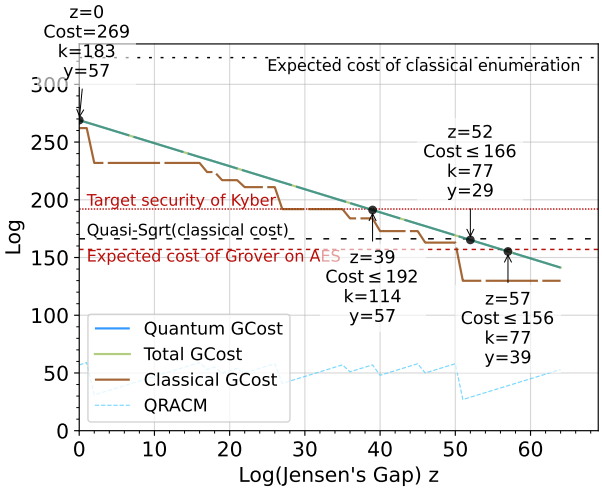
Figure 13: Cost estimation for Kyber with the instantiation for operator $\mathcal{W}$ as in § 4.1 and with DF = 1, QD = 1, $b = 1$ (see § 3).



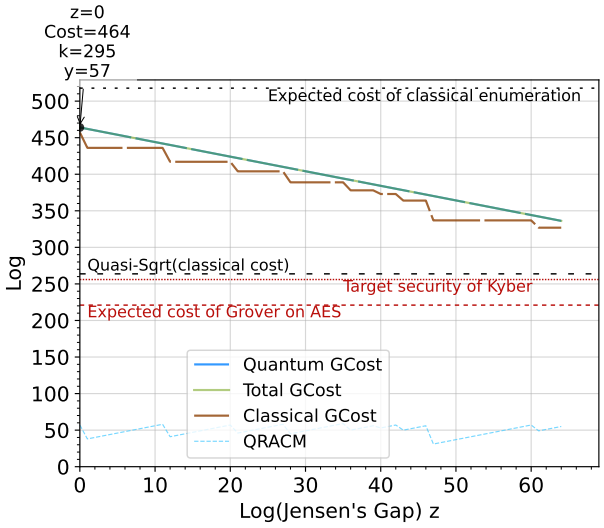
(a) Kyber-768, MaxDepth $= 2^{40}$



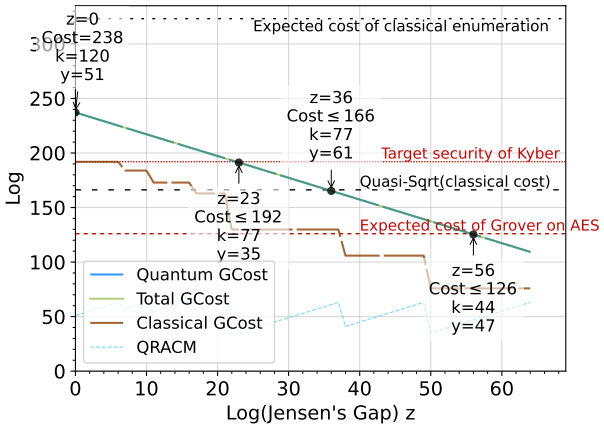
(b) Kyber-1024, MaxDepth $= 2^{40}$.



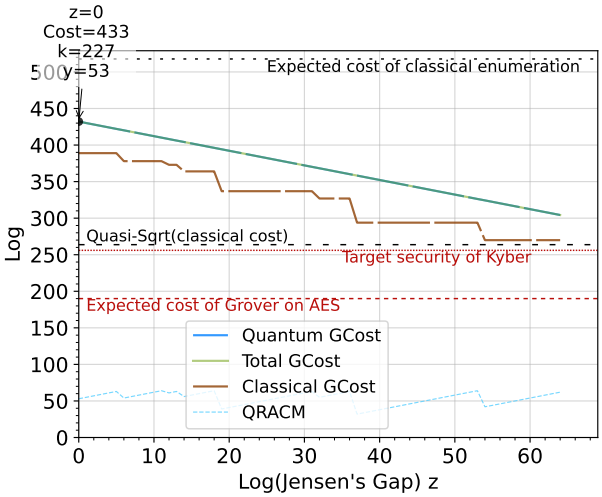
(c) Kyber-768, MaxDepth $= 2^{64}$



(d) Kyber-1024, MaxDepth $= 2^{64}$.



(e) Kyber-768, MaxDepth $= 2^{96}$



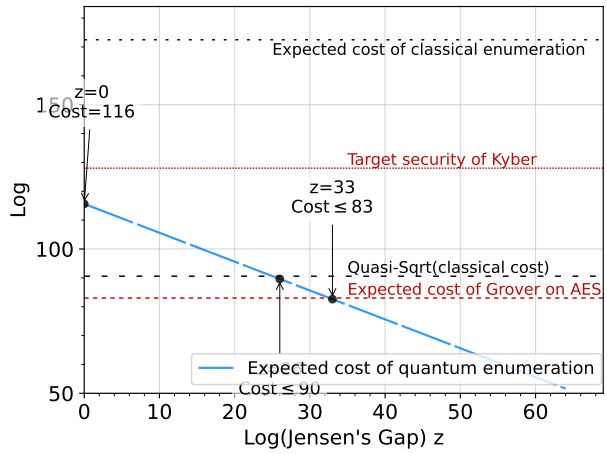
(f) Kyber-1024, MaxDepth $= 2^{96}$.

Figure 14: Cost estimation for Kyber-768 under different MaxDepth restrictions with the instantiation for operator $\mathcal{W}$ as in § 4.1 and with DF $= 1$, QD $= 1$, $b = 1$ (see § 3).

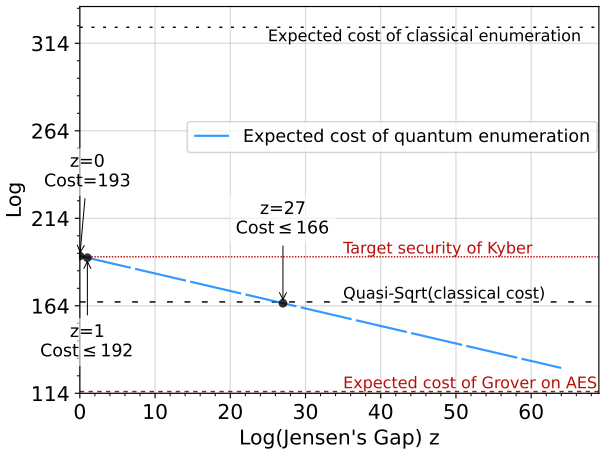
E.2 Figures for the Circuit-based Results

In this section we present the figures from our estimations of the cost of quantum enumeration that we have not reported in § 5. For all cost estimations the quantum operator GCost($\mathcal{W}$) and TDepth($\mathcal{W}$) are estimated as in § 4.2 and with $DF(\mathcal{W})$, $QD(\mathcal{W})$, $WQ(\mathcal{T}, W)$ as in § 3,

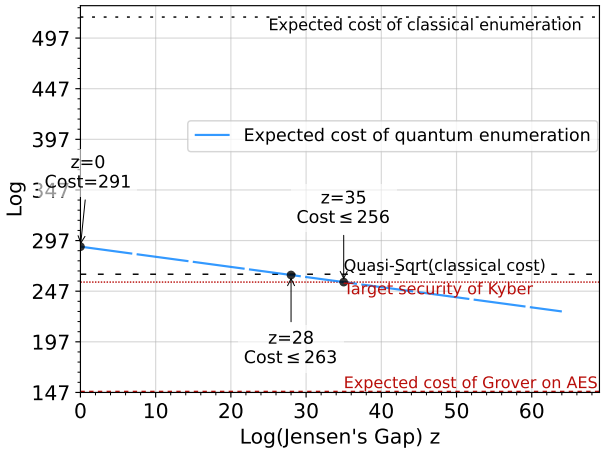
- Figure 15 shows the cost estimation without any MaxDepth constraint on T-Depth(QPE($\mathcal{W}$))
- Figure 16 is the cost estimation for Kyber-512 with T-Depth(QPE($\mathcal{W}$)) $\leq 2^{64}$
- Figure 17 shows the cost estimation for Kyber-768 and Kyber-1024 with T-Depth(QPE($\mathcal{W}$)) $\leq \{2^{40}, 2^{64}, 2^{96}\}$ respectively.



(a) Cost estimation for Kyber-512 without MaxDepth restrictions.

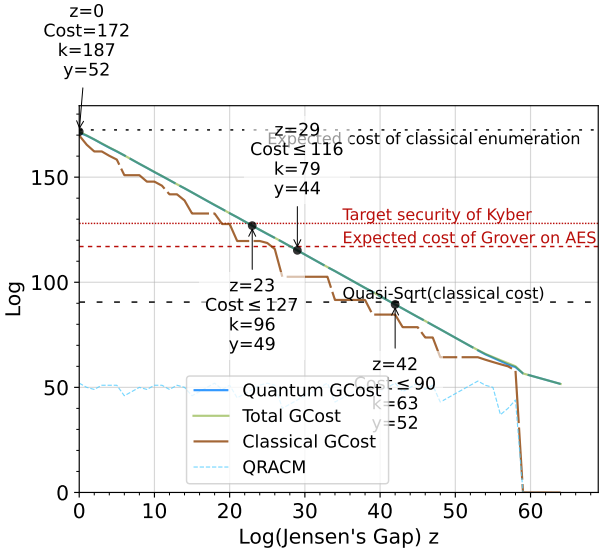


(b) Cost estimation for Kyber-768 without MaxDepth restrictions.

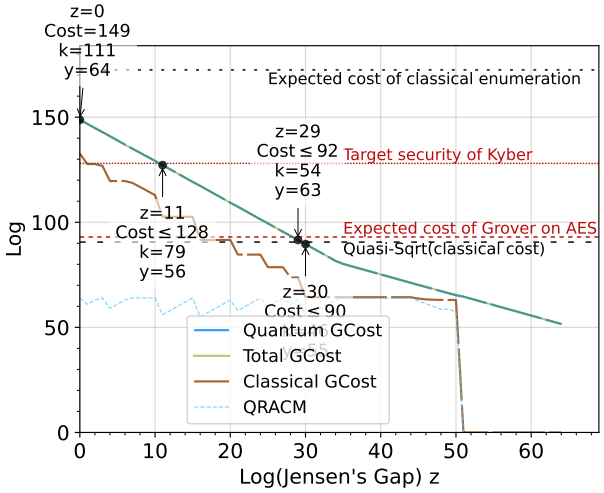


(c) Cost estimation for Kyber-1024 without MaxDepth restrictions.

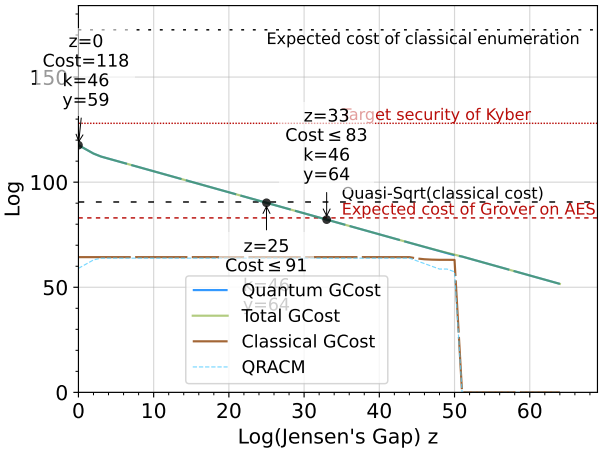
Figure 15: Cost estimation for Kyber with no MaxDepth restriction and the instantiation for operator $\mathcal{W}$ as in § 4.2 and with DF = 1, QD = 1, $b = 1$ (see § 3).



(a) Kyber-512, MaxDepth $= 2^{40}$

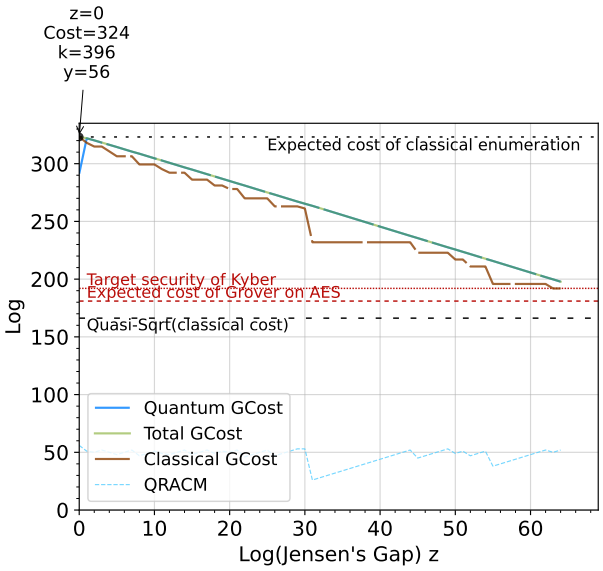


(b) Kyber-512, MaxDepth $= 2^{64}$

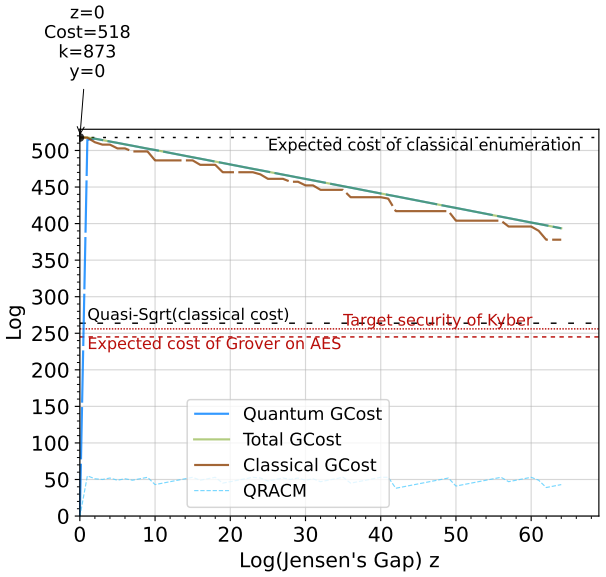


(c) Kyber-512, MaxDepth $= 2^{96}$

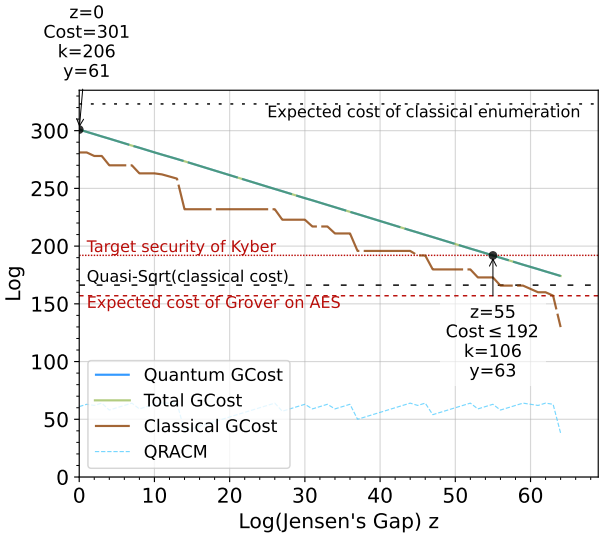
Figure 16: Cost estimation for Kyber-512 under different MaxDepth restrictions with the instantiation for operator $\mathcal{W}$ as in § 4.2 and with DF $= 1$, QD $= 1$, $b = 1$ (see § 3).



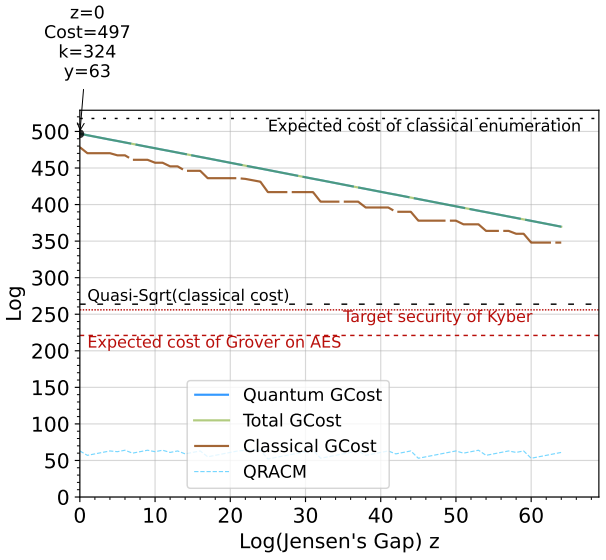
(a) Kyber-768, MaxDepth $= 2^{40}$



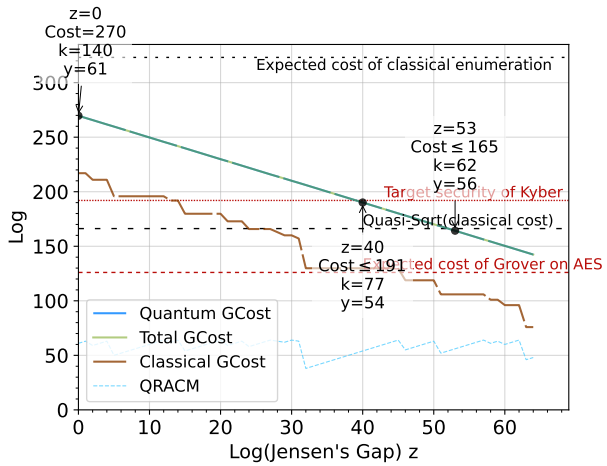
(b) Kyber-1024, MaxDepth $= 2^{40}$.



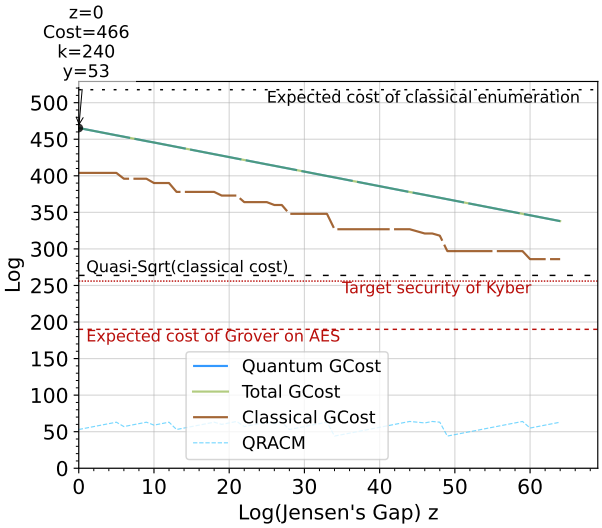
(c) Kyber-768, MaxDepth $= 2^{64}$



(d) Kyber-1024, MaxDepth $= 2^{64}$.



(e) Kyber-768, MaxDepth $= 2^{96}$



(f) Kyber-1024, MaxDepth $= 2^{96}$.

Figure 17: Cost estimation for Kyber-768 and Kyber-1024 under different MaxDepth restrictions with the instantiation for operator $\mathcal{W}$ as in § 4.2 and with DF $= 1$, QD $= 1$, $b = 1$ (see § 3).

F Beyond Lower Bounds for DF, QD, WQ

We recall in Fig. 18 the quantities representing the number of calls of the FindMV, DetectMV, quantum phase estimation and $\mathcal{W}$ procedures, such that the depth and gate-cost of FindMV amounts to

$$\begin{aligned}\text{T-Depth}(\text{FindMV}(\mathcal{T})) &= \text{DF}(\mathcal{T}) \cdot \text{QD}(\mathcal{T}) \cdot \text{WQ}(\mathcal{T}, \mathcal{W}) \cdot \text{T-Depth}(\mathcal{W}) \\ \text{GCost}(\text{FindMV}(\mathcal{T})) &= \text{DF}(\mathcal{T}) \cdot \text{QD}(\mathcal{T}) \cdot \text{WQ}(\mathcal{T}, \mathcal{W}) \cdot \text{GCost}(\mathcal{W}).\end{aligned}$$

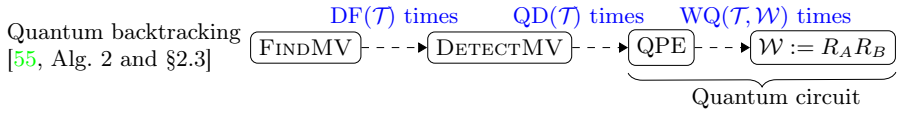


Figure 18: Overview of Montanaro's marked vertex (MV) finding backtracking algorithm [55]. Only part of the algorithm (QPE) needs to run within MaxDepth.

Let the tree $\mathcal{T}$ being searched be of depth h and the degree of each node be bound by $\mathcal{C}$.

DF($\mathcal{T}$). As mentioned in § 3.1, the analysis in [13] assumes an implicit transformation of $\mathcal{T}$ into a binary tree of depth $h \log \mathcal{C}$. DetectMV is then called on the root level, in order to detect whether marked vertices are in the tree. If no marked vertices are found, no more calls are required, and $\text{DF}(\mathcal{T}) = 1$. If there are, further calls are made to identify the path from the root to the marked leaf, akin to a binary search. In total, in order to identify one marked leaf in $\mathcal{T}$ (or return error),

$$\text{DF}(\mathcal{T}) = \begin{cases} 1, & \mathcal{T} \text{ contains no marked leaves,} \\ h \log \mathcal{C} & \text{contains at least one marked leaf.} \end{cases}$$

In the setting of combined classical-quantum enumeration, most of the $H_k/2^y$ trees $\mathcal{T}(g)$ explored will not contain any marked leafs. Given how the quantum GCost is estimated in (6) as

$$\frac{H_k}{2^y} \cdot \mathbb{E}[\text{GCost}(\text{FindMV}(\mathcal{T}(g)))] = \frac{H_k}{2^y} \cdot \mathbb{E}[\text{DF}(\mathcal{T}(g)) \cdot \text{QD}(\mathcal{T}(g)) \cdot \text{WQ}(\mathcal{T}(g), \mathcal{W}) \cdot \text{GCost}(\mathcal{W})],$$

we could set $\text{DF}(\mathcal{T}(g)) = \left(\frac{H_k}{2^y} - 1 + h \log \mathcal{C}\right) \cdot \frac{2^y}{H_k}$ with $h = n - k + 1$ during cost estimation, to capture how when the enumeration radius is short enough, $h \log \mathcal{C}$ calls will likely be made on only one of the subtrees, and one call will be made to the remaining $\frac{H_k}{2^y} - 1$ many subtrees.

This discussion implicitly assumes the correctness of DetectMV, which however can return incorrect results, increasing the complexity of an estimation on DF. One option would be running DetectMV multiple times per level,

with the amount of repetition depending on analysis of the specific DetectMV implementation, as well as the noise model of the quantum computer.

A simpler analysis would be to estimate the failure probability of FindMV when calling DetectMV once per level, assuming a tight failure probability upper bound for DetectMV($\mathcal{T}$) of δ_{DMV}. Then, the success probability of FindMV would be about $(1-\delta_{\mathrm{DMV}})^{\mathrm{DF}(\mathcal{T})}$, which is $\mathcal{O}(1)$ if $\delta_{\mathrm{DMV}} \approx 1/\mathrm{DF}(\mathcal{T})$.[13] Since *a priori* we don't know whether $\mathcal{T}$ contains a marked vertex, this would mean implementing DetectMV such that $\delta_{\mathrm{DMV}} \approx (h \log \mathcal{C})^{-1}$, to account for the in-principle $h \log \mathcal{C}$ successful calls required to detect the marked vertex in its subtree. We proceed to do this next.

QD($\mathcal{T}$). The way DetectMV [55, Alg. 2] is computed is by performing multiple times, say K many, quantum phase estimation (QPE) on the $\mathcal{W}$ operator.

Let X_i be a random variable valued 1 when the i^{th} call to $QPE(\mathcal{W})$ returned an eigenvalue 1, and valued 0 otherwise, and let $Y_K = \sum_{i\in[K]} X_i$. The X_i are then *i.i.d.* Bernoulli random variables. Let MV denote the event that a marked vertex is contained in $\mathcal{T}$ and $\overline{MV}$ the opposite event. The core idea around DetectMV is that whenever a marked vertex exists, $QPE(\mathcal{W})$ will tend to return 1, and 0 otherwise. Indeed, from [55, Proof of Lemma 2.4] we have that $p_1 \coloneqq \Pr[X_i = 1 \mid \overline{MV}] \leq 1/4$ and $p_2 \coloneqq \Pr[X_i = 0 \mid MV] \leq 1/2$. In order to decide whether a tree contains a marked vertex, we run K instances of QPE on $\mathcal{W}$, and then check whether enough instances returned 1: for some fixed $\alpha \in (0, 1]$ to be determined, we return "marked vertex exists" if and only if $Y_K \geq \alpha K$.

As seen in the previous paragraph on DF($\mathcal{T}$), we may want to fix a target failure probability δ_{DMV} for DetectMV, which can be achieved by picking K high enough. We start assuming we have found α, and use Chernoff bounds to estimate upper bounds on the "false positive/negative" probabilities $\Pr[Y \geq \alpha K \mid \overline{MV}]$ and $\Pr[Y \leq \alpha K \mid MV]$. By direct computation of the bound, recalling that the X_i are *i.i.d.*, we have

$$\begin{aligned}\Pr[Y_K \geq \alpha K \mid \overline{MV}] &\leq \exp(-t\alpha K)\mathbb{E}\left[\exp(tY_K)\right] = \exp(-t\alpha K)\mathbb{E}\left[\prod_{i=1}^{K}\exp(tX_i)\right] \\ &= \exp(-t\alpha K)\mathbb{E}\left[(\exp(tX_1))\right]^K = \left(\frac{1+p_1(\exp(t)-1)}{\exp(\alpha t)}\right)^K,\end{aligned}$$

for any $t > 0$. For $\Pr[Y_K \leq \alpha K]$ a similar computation on the left tail gives

$$\Pr[Y_K \leq \alpha K \mid MV] \leq \left(\frac{\exp(t)+p_2(1-\exp(t))}{\exp(\alpha t)}\right)^K, \text{ for any } t < 0.$$

We then identify values of α and ε such that $\inf_{y>0}\left(\frac{1+p_1(\exp(t)-1)}{\exp(\alpha t)}\right)^{\varepsilon} \leq 1/2$ and $\inf_{y<0}\left(\frac{\exp(t)+p_2(1-\exp(t))}{\exp(\alpha t)}\right)^{\varepsilon} \leq 1/2$. From a numerical search, and picking

[13] Specifically, it approaches e^{-1} as $\delta_{\mathrm{DMV}} \to 0$.

the smallest possible ε returned, we observe a valid pair at $\alpha = 0.369017$ and $\varepsilon = 20$.[14]

Finally, we compute

$$\Pr[Y_K \geq \alpha K \mid \overline{MV}] \leq \inf_{y>0} \left(\frac{1 + p_1(\exp(t) - 1)}{\exp(\alpha t)} \right)^K \leq (1/2)^{K/\varepsilon},$$

and similarly for $\Pr[Y_K \leq \alpha K \mid MV]$, suggesting that to get an overall failure probability of at most δ_{DMV}, one should choose $K/\varepsilon \geq \log(1/\delta_{\mathrm{DMV}})$. Therefore, it should be sufficient to choose

$$\mathrm{QD}(\mathcal{T}) = K = \lceil 20 \log(h \log \mathcal{C}) \rceil \geq \varepsilon \log(1/\delta_{\mathrm{DMV}}).$$

$WQ(\mathcal{T}, \mathcal{W})$. As used inside of DetectMV, QPE needs to be run with precision $b/\sqrt{\#\mathcal{T} \cdot h}$, for some constant $b > 0$, returning after $\approx \sqrt{\#\mathcal{T} \cdot h}/b$ evaluations of $\mathcal{W}$. In [19, Sec 4.1], it is shown that the constant $b > 0$ can be bound using $p_1 \leq 2\sqrt{b}(1 + o(1))$ where $p_1 := \Pr[X_i = 1 \mid \overline{MV}]$, and setting p_1 as low as possible. In our case, this is $p_1 \leq 1/4$, which implies using at least $b \geq 1/64$. Since we require evaluating $\mathcal{W}$ approximately $\approx \sqrt{\#\mathcal{T} \cdot h}/b$ times and $1/b \leq 64$, to be safe we would need to pick

$$\mathrm{WQ}(\mathcal{T}, \mathcal{W}) \approx 64\sqrt{\#\mathcal{T} \cdot h}\,.$$

F.1 Results from More Bounds

We replicate the analysis performed in § 5.2, estimating the cost of a combined classical-quantum attack as described in § 3.2 and 3.3. In Tab. 9 we summarize the values for the Jensen's gap 2^z at crossover points of our combined classical-quantum enumeration attacks against Kyber and different "success metrics" to claim an attack.[15] The cost of Grover's search against AES is taken from [39, Tables 10 and 12]. Tab. 9 corresponds to attacks in the settings for $\mathcal{W}$ as in § 4.1 and § 4.2 with $DF(\mathcal{W})$, $QD(\mathcal{W})$, $WQ(\mathcal{T}, W)$ as in App. F.

Overall, comparing Tab. 9 with Tab. 6 it would appear that in terms of increased costs, compared to a setting with strict lower bounds (§ 3.1) in the query-based model of $\mathcal{W}$, using the results from App. F in the query-based model increases costs less than using the circuit-based model and the lower bounds from § 3.1.

[14] In particular, our value of α is quite close to the $3/8 = 0.375$ proposed in [55]. This latter value would however imply $\varepsilon = 22$.

[15] We remark that exact crossovers happen at fractional values of z. In these tables we take the smallest value z such that the total cost is less than the corresponding quantity and give the level k for which this is achieved.

Table 9: Summary of the values for the Jensen's gap 2^z at crossover points of our combined classical-quantum enumeration attacks against Kyber and different "success metrics" to claim an attack. We remark that exact crossovers happen at fractional values of z. In this table we round down threshold values of z. The cost of Grover's search against AES is taken from [39, Tables 10 and 12]. The results are estimated from the bounds $\mathrm{DF}(\mathcal{W}) = 1$, $\mathrm{QD}(\mathcal{W}) = \lceil 20 \log(h \log \mathcal{C}) \rceil$, $b = 1/64$.

more likely to be feasible — less likely to be feasible

MD	Kyber	$\log \mathbb{E}[\text{GCost}]$ (with $\mathcal{W}$ as in § 4.1) below… Target security	Grover on $\text{AES}_{\{128,192,256\}}$	Quasi-Sqrt $\sqrt[1/b]{\#\mathcal{T} \cdot h}$	$\log \mathbb{E}[\text{GCost}]$ (with $\mathcal{W}$ as in § 4.2) below… Target security	Grover on $\text{AES}_{\{128,192,256\}}$	Quasi-Sqrt $\sqrt[1/b]{\#\mathcal{T} \cdot h}$
2^{40}	-512	$z \geq 13,\ k \leq 92$	$z \geq 19,\ k \leq 83$	$z \geq 32,\ k \leq 59$	$z \geq 29,\ k \leq 96$	$z \geq 35,\ k \leq 79$	$z \geq 48,\ k \leq 63$
	-768	$z \geq 57,\ k \leq 114$	$z \geq 63,\ k \leq 106$	$z > 64$	$z > 64$	$z > 64$	$z > 64$
	-1024	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$
2^{64}	-768	$z \geq 45,\ k \leq 114$	$z \geq 63,\ k \leq 77$	$z \geq 58,\ k \leq 77$	$z \geq 61,\ k \leq 106$	$z > 64$	$z > 64$
	-512	$z \geq 1,\ k \leq 92$	$z \geq 19,\ k \leq 64$	$z \geq 20,\ k \leq 59$	$z \geq 17,\ k \leq 79$	$z \geq 35,\ k \leq 54$	$z \geq 36,\ k \leq 46$
	-1024	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$
2^{96}	-512	$z \geq 0,\ k \leq 40$	$z \geq 14,\ k \leq 46$	$z \geq 7,\ k \leq 46$	$z \geq 1,\ k \leq 58$	$z \geq 39,\ k \leq 46$	$z \geq 31,\ k \leq 46$
	-768	$z \geq 29,\ k \leq 77$	$z \geq 62,\ k \leq 44$	$z \geq 42,\ k \leq 77$	$z \geq 46,\ k \leq 77$	$z > 64$	$z \geq 59,\ k \leq 62$
	-1024	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$
∞	-512	$z \geq 0,\ k = 0$	$z \geq 15,\ k = 0$	$z \geq 7,\ k = 0$	$z \geq 0,\ k = 0$	$z \geq 39,\ k = 0$	$z \geq 32,\ k = 0$
	-768	$z \geq 0,\ k = 0$	$z \geq 58,\ k = 0$	$z \geq 7,\ k = 0$	$z \geq 7,\ k = 0$	$z > 64$	$z \geq 33,\ k = 0$
	-1024	$z \geq 15,\ k = 0$	$z > 64$	$z \geq 7,\ k = 0$	$z \geq 41,\ k = 0$	$z > 64$	$z \geq 34,\ k = 0$

G Lower-Bounding Additive and Multiplicative Jensen Gaps

Let X be the random variable for $\#\mathcal{T}$ with support $\text{supp}(X) \subset [1, +\infty)$, $\mu = \mathbb{E}[X]$, $\sigma^2 = \mathbb{V}[X]$. In this section, we explore two different derivations for lower bounds of the additive $(\sqrt{\mathbb{E}[X]} - \mathbb{E}[\sqrt{X}])$ and multiplicative $(\sqrt{\mathbb{E}[X]}/\mathbb{E}[\sqrt{X}])$ Jensen's gaps of X.

As part of our derivations, we will use the notion of Hölder continuity, which we recall.

Definition 2 (Hölder continuity). *A real function $f\colon \mathbb{R} \to \mathbb{R}$ is α-Hölder continuous over an interval $(a, b) \subset \mathbb{R}$ if for any $x, y \in (a, b)$, there exists $M > 0$ such that $|f(x) - f(y)| \leq M \cdot |x - y|^{\alpha}$.*

G.1 Bounding the Additive Jensen's Gap.

We start by proving that the square root function $\sqrt{x}$ is $\frac{1}{2}$-Hölder continuous.

Lemma 3. *$\sqrt{x}$ is $\frac{1}{2}$-Hölder continuous over $(0, \infty)$ with $M \geq 1$.*

Proof. Let $x, y \geq 0$. First we write

$$|\sqrt{x} - \sqrt{y}| = \frac{|\sqrt{x} - \sqrt{y}| \cdot |\sqrt{x} + \sqrt{y}|}{|\sqrt{x} + \sqrt{y}|} = \frac{|x - y|}{\sqrt{x} + \sqrt{y}} = \sqrt{|x - y|} \cdot \frac{\sqrt{|x - y|}}{\sqrt{x} + \sqrt{y}}.$$

Now, note that

$$|x - y| \leq |x| + |y| = x + y \leq x + y + 2\sqrt{xy} = (\sqrt{x} + \sqrt{y})^2$$

Since $\sqrt{x}$ is strictly increasing over $(0, \infty)$,

$$|x - y| \leq (\sqrt{x} + \sqrt{y})^2 \iff \sqrt{|x - y|} \leq \sqrt{x} + \sqrt{y}. \iff \frac{\sqrt{|x - y|}}{\sqrt{x} + \sqrt{y}} \leq 1.$$

Hence,

$$|\sqrt{x} - \sqrt{y}| = \sqrt{|x - y|} \cdot \frac{\sqrt{|x - y|}}{\sqrt{x} + \sqrt{y}} \leq \sqrt{|x - y|},$$

which concludes the proof. □

Lemma 4. *The additive Jensen gap of X is in absolute value $|\mathbb{E}[\sqrt{X}] - \sqrt{\mu}| \leq \sqrt{\sigma}$.*

Proof. Following [33, Eq 1.1],

$$
\begin{aligned}
|\mathbb{E}[\sqrt{X}] - \sqrt{\mu}| &= \left| \sum_{x \in \mathrm{supp}(X)} \sqrt{x} \Pr[X = x] - \sqrt{\mu} \right| \\
&= \left| \sum_{x \in \mathrm{supp}(X)} (\sqrt{x} - \sqrt{\mu}) \Pr[X = x] \right| \quad (\text{by } \sum_{x \in \mathrm{supp}(X)} \Pr[X = x] = 1) \\
&\leq \sum_{x \in \mathrm{supp}(X)} \left| \sqrt{x} - \sqrt{\mu} \right| \Pr[X = x] \\
&\leq \sum_{x \in \mathrm{supp}(X)} \sqrt{|x - \mu|} \Pr[X = x] \qquad (\text{by Lem. 3}) \\
&= \mathbb{E}\left[\sqrt{|x - \mu|} \right] \qquad (\text{by definition of } \mathbb{E}) \\
&\leq \sqrt{\mathbb{E}\left[|x - \mu| \right]} \qquad (\text{by Jensen's inequality}) \\
&\leq \mathbb{E}\left[(x - \mu)^2 \right]^{1/4} \qquad (\text{by Jensen's inequality}) \\
&= \sqrt{\sigma}. \qquad (\text{by definition of } \sigma)
\end{aligned}
$$

This concludes the proof. □

Lemma 5. $\mathbb{E}[\textsc{GCost}(QPE(\mathcal{W}))] \geq \sqrt{h}\left(\sqrt{\mu} - \sqrt{\sigma}\right) \cdot \textsc{GCost}(\mathcal{W})$, *where* h *is the height of* $\mathcal{T}(g_r)$.

Proof. From Lem. 4,

$$
\begin{aligned}
|\mathbb{E}[\sqrt{X}] - \sqrt{\mu}| \leq \sqrt{\sigma} &\iff -\sqrt{\sigma} \leq \mathbb{E}[\sqrt{X}] - \sqrt{\mu} \leq \sqrt{\sigma} \\
&\implies \mathbb{E}[\sqrt{X}] \geq \sqrt{\mu} - \sqrt{\sigma}.
\end{aligned}
$$

Combine with

$$\textsc{GCost}(\mathrm{QPE}(\mathcal{W})) = \mathrm{WQ}(\mathcal{T}(g_r), \mathcal{W}) \cdot \textsc{GCost}(\mathcal{W}),$$

and with Simpl. Assm. 1, letting $X = \#\mathcal{T}(g_r)$. □

G.2 Bounding the Multiplicative Jensen's Gap.

We start by proving that the natural logarithm function $\ln(x)$ is $\frac{1}{2}$-Hölder continuous and that $\ln \sqrt{x}$ is $\frac{1}{2}$-Hölder continuous.

Lemma 6. $\ln(x)$ *is* $\frac{1}{2}$*-Hölder continuous over* $[1, \infty)$ *with* $M \geq 1$.

Proof. Let $p = 2$. Noting $\frac{d}{dx} \ln x = \frac{1}{x}$, and that $x, y \geq 1$

$$
\begin{aligned}
|\ln x - \ln y| = \left| \int_x^y \frac{dt}{t} \right| &\leq \int_{\min(x,y)}^{\max(x,y)} \left| \frac{1}{t} \right| dt \\
&\leq \left(\int_{\min(x,y)}^{\max(x,y)} 1^2 dt \right)^{1/2} \left(\int_{\min(x,y)}^{\max(x,y)} \left| \frac{1}{t} \right|^2 dt \right)^{1/2} && \text{(by Hölder's inequality)} \\
&= |y - x|^{1/2} \left(\int_{\min(x,y)}^{\max(x,y)} \frac{1}{t^2} dt \right)^{1/2} \\
&\leq |y - x|^{1/2} \left(\int_1^{\infty} \frac{1}{t^2} dt \right)^{1/2} \leq |y - x|^{1/2} \cdot \left[-\frac{1}{x} \right]_1^{\infty} = |y - x|^{1/2}.
\end{aligned}
$$

□

Lemma 7. *$\ln \sqrt{x}$ is $\frac{1}{2}$-Hölder continuous over $[1, \infty)$ with $M \geq 1/2$.*

Proof.

$$
|\ln \sqrt{x} - \ln \sqrt{y}| = \frac{1}{2} |\ln x - \ln y| \leq \frac{1}{2} |x - y|^{1/2},
$$

where the second inequality follows from Lem. 6. □

Lemma 8. *Given a multiplicative Jensen's gap 2^z such that $\mathbb{E}[\sqrt{x}] = 2^{-z} \sqrt{\mathbb{E}[x]}$, then $z \leq \frac{1}{2 \ln 2} \sqrt{\sigma}$.*

Proof. First we take logarithms,

$$
\mathbb{E}[\sqrt{x}] = 2^{-z} \sqrt{\mathbb{E}[x]} \iff \ln \mathbb{E}[\sqrt{x}] = -z \ln 2 + \ln \sqrt{\mu}.
$$

Then we upper bound z as

$$
\begin{aligned}
z &= \frac{1}{\ln 2} (\ln \sqrt{\mu} - \ln \mathbb{E}[\sqrt{x}]) \\
&\leq \frac{1}{\ln 2} (\ln \sqrt{\mu} - \mathbb{E}[\ln \sqrt{x}]) && \text{(by Jensen's inequality, } \mathbb{E} \ln \leq \ln \mathbb{E}) \\
&= \frac{1}{\ln 2} \sum_{\text{supp}(X)} \left(\ln \sqrt{\mu} - \ln \sqrt{x} \right) \Pr[X = x] \\
&\leq \frac{1}{\ln 2} \sum_{\text{supp}(X)} \left| \ln \sqrt{\mu} - \ln \sqrt{x} \right| \Pr[X = x] \\
&\leq \frac{1}{\ln 2} \sum_{\text{supp}(X)} \frac{1}{2} \left| \mu - x \right|^{1/2} \Pr[X = x] && \text{(by Lem. 7)} \\
&= \frac{1}{2 \ln 2} \mathbb{E}[|\mu - x|^{1/2}] && \text{(by definition of } \mathbb{E}) \\
&\leq \frac{1}{2 \ln 2} \mathbb{E}\left[(x - \mu)^2 \right]^{1/4} = \frac{1}{2 \ln 2} \sqrt{\sigma} && \text{(by applying Jensen's inequality twice)}
\end{aligned}
$$

□

Lemma 9. $\mathbb{E}[\mathrm{GCost}(QPE(\mathcal{W}))] \geq 2^{-\frac{1}{2\ln 2}\sigma}\sqrt{\mu \cdot h} \cdot \mathrm{GCost}(\mathcal{W})$, *where* h *is the height of* $\mathcal{T}(g_r)$.

Proof. Using Lem. 8,

$$\mathbb{E}[\sqrt{X}] = 2^{-z}\sqrt{\mu} \geq 2^{-\frac{1}{2\ln 2}\sqrt{\sigma}}\sqrt{\mu}.$$

Combine with

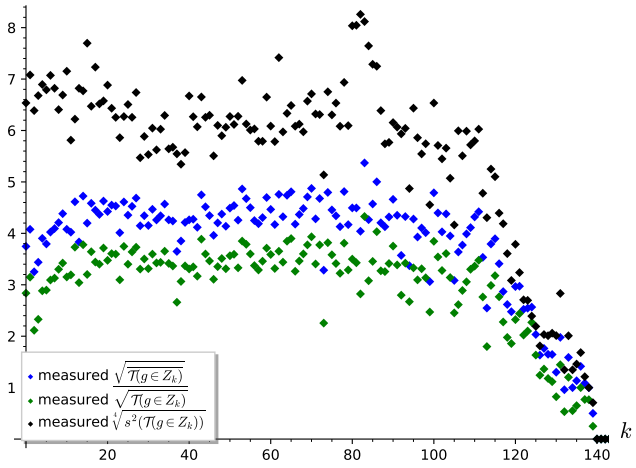
$$\mathrm{GCost}(\mathrm{QPE}(\mathcal{W})) = \mathrm{WQ}(\mathcal{T}(g_r), \mathcal{W}) \cdot \mathrm{GCost}(\mathcal{W}),$$

and with Simpl. Assm. 1, letting $X = \#\mathcal{T}(g_r)$. □

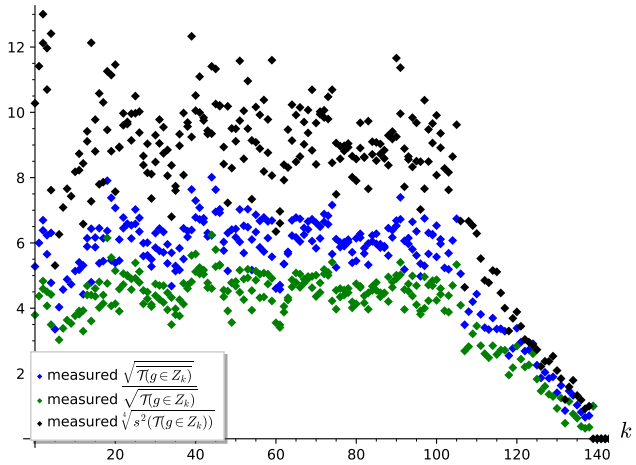
G.3 Experimentally Estimating $\sqrt{\sigma}$

Unfortunately, we are not aware of any techniques to estimate the variance of $\#\mathcal{T}(g)$ for an element $g \in Z_k$ of a pruned enumeration tree. Therefore, we have implemented an extension to the experiments from App. C originally used to experimentally measure some multiplicative Jensen's gaps. This extension lets us measure the unbiased sample variance s^2 of the number of nodes $\#\mathcal{T}(g)$. Using s^2 as an estimate for the variance σ^2 of the distribution, we then plot $\sqrt[4]{s^2}$ as an estimate for $\sqrt{\sigma}$ in Figs. 19 and 20.

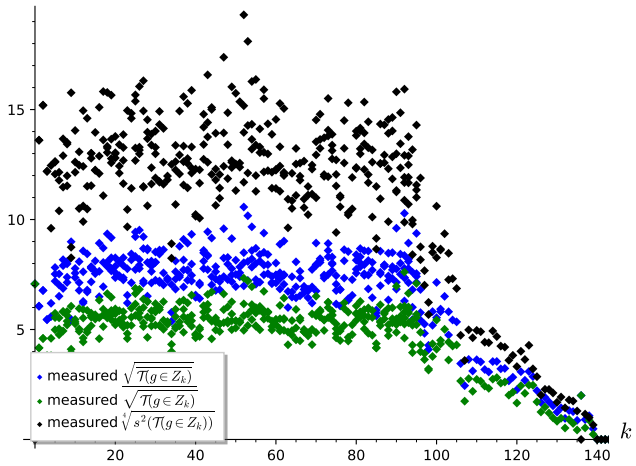
Our results seem to indicate that $\sqrt{\sigma}$ increases with the dimension of the tree β and with the level k where g is chosen, suggesting that shallower subtrees are more exposed to variations in their size compared to deeper trees. The value of the measured estimates $\sqrt[4]{s^2}$ seems also to stay relatively low (in absolute value) in our experiments, peaking at around 20. In any case, these experiments are no more or less conclusive than those in App. C, and hence this analysis does not seem to significantly improve on the methodology used in § 5.2 for estimating the cost of the attack, that is testing multiple values of possible Jensen's gaps.



(a) $n = 72$ (cf. Tab. 7), block size $\beta = 50$, 10^{th} tour, $h = 20$.

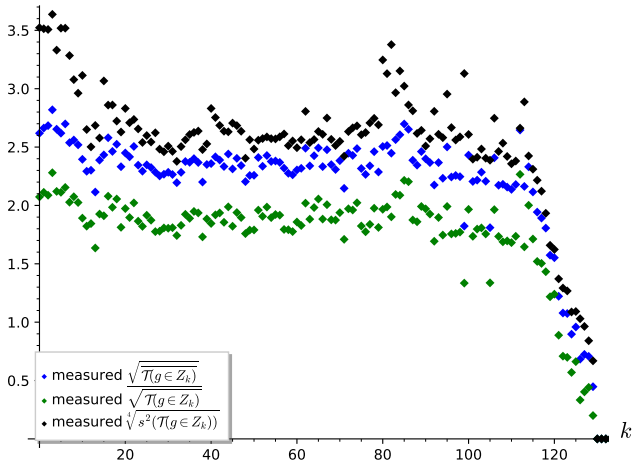


(b) $n = 72$ (cf. Tab. 7), block size $\beta = 60$, 10^{th} tour, $h = 20$.

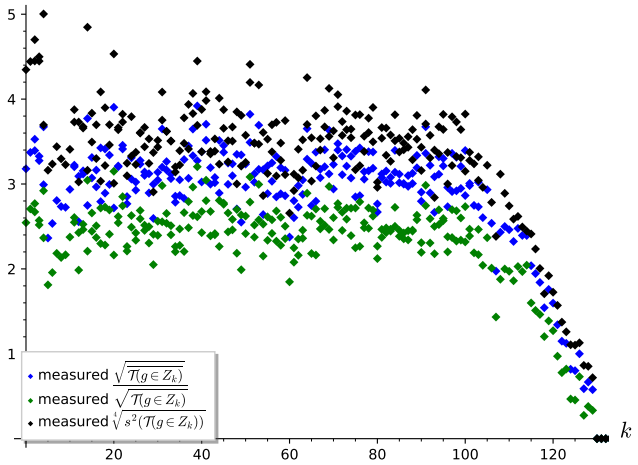


(c) $n = 72$ (cf. Tab. 7), block size $\beta = 70$, 10^{th} tour, $h = 20$.

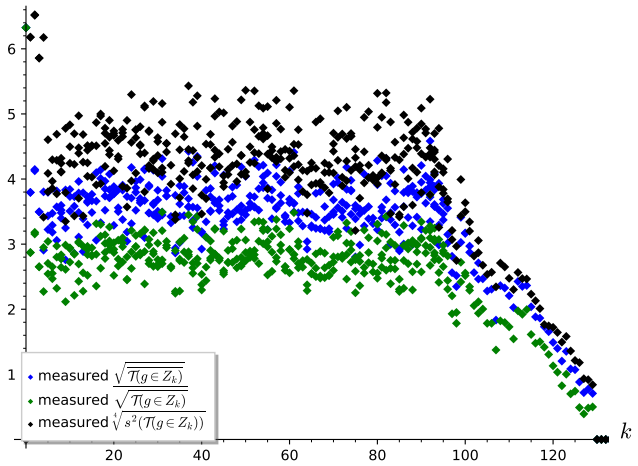
Figure 19: Measurement on the Jensen gap of subtrees of height $h = 20$ in pruned enumeration experiments. The height h is reduced in the last few blocks of the basis of dimension smaller than h. Blue dots are the squared root of the average, green dots are the average of the squared root.



(a) $n = 72$ (cf. Tab. 7), block size $\beta = 50$, 10^{th} tour, $h = 30$.



(b) $n = 72$ (cf. Tab. 7), block size $\beta = 60$, 10^{th} tour, $h = 30$.



(c) $n = 72$ (cf. Tab. 7), block size $\beta = 70$, 10^{th} tour, $h = 30$.

Figure 20: Measurement on the Jensen gap of subtrees of height $h = 30$ in pruned enumeration experiments. The height h is reduced in the last few blocks of the basis of dimension smaller than h. Blue dots are the squared root of the average, green dots are the average of the squared root.