Sampling Arbitrary Discrete Distributions for RV Commitment Schemes Using the Trimmed-Tree Knuth-Yao Algorithm

Zoë Ruha Bell
University of California, Berkeley

Anvith Thudi
University of Toronto

May 8, 2025

Abstract

Sampling from non-uniform randomness according to an algorithm which keeps the internal randomness used by the sampler hidden is increasingly important for cryptographic applications, such as timing-attack-resistant lattice-based cryptography or certified differential privacy. In this paper we present a provably efficient sampler that maintains random sample privacy, or *random sample hiding*, and is applicable to arbitrary discrete random variables. Namely, we present a constant-time version of the classic Knuth-Yao algorithm that we name "trimmed-tree" Knuth-Yao. We establish distribution-tailored Boolean circuit complexity bounds for this algorithm, in contrast to the previous nïave distribution-agnostic bounds. For a σ^2-sub-Gaussian discrete distribution where b_t is the number of bits for representing the domain, and b_p is the bits for precision of the PDF values, we prove the Boolean circuit complexity of the trimmed-tree Knuth-Yao algorithm has upper bound $O(\sigma b_p^{3/2} b_t)$, an exponential improvement over the naïve bounds, and in certain parameter regimes establish the lower bound $\widetilde{\Omega}((\sigma + b_p)b_t)$. Moreover, by proving the subtrees in the trimmed-tree Knuth-Yao circuit are small, we prove it can computed by running b_p circuits of size $O(\sigma b_p^{1/2} b_t)$ in parallel and then running $O(b_p b_t)$ sequential operations on the output. We apply these circuits for trimmed-tree Knuth-Yao to constructing random variable commitment schemes for arbitrary discrete distributions, giving exponential improvements in the number of random bits and circuit complexity used for certified differentially private means and counting queries over large datasets and domains.

Keywords: Discrete Samplers, Verifiable Computation, Differential Privacy

1 Introduction

Randomization is a key tool for many cryptographic applications. Specifically, in an increasing number of cases, there is a need to sample complex, non-uniform randomness, such as in privacy-protected machine learning, differential privacy, and lattice-based cryptography. Further, for some of these applications, it is necessary to prove that this randomness was sampled according to a particular distribution, such as the discrete Gaussian, while hiding which element of the support was sampled. This allows for constructions which, for instance, protect against timing attacks or provide zero knowledge for random variable commitment schemes. Towards these ends, our goal in this work is to correctly and efficiently sample from probability distributions such as the discrete Gaussian while maintaining random sample privacy.

To do so, we turn to the classic Knuth-Yao algorithm, which uses a discrete distribution generation tree, or DDG tree, as a data structure for sampling from a specific distribution while using an essentially minimal number of random bits (≤ 2 more than the entropy of the distribution). However, the standard Knuth-Yao algorithm is not random-sample private, as the length of the path that was taken through the DDG tree leaks information about which element of the support was chosen. In order to achieve random sample privacy, we consider a variant of Knuth-Yao we name "trimmed-tree Knuth-Yao." To the best of our knowledge, constant-time versions of Knuth-Yao were used in past-work without full specification of either the standard or modified Knuth-Yao algorithms, often primarily as a point of comparison for other approaches. One of our major "non-technical" contributions is recalling in full detail the Knuth-Yao algorithm and the DDG

tree data structure it is based on, and then fully specifying what we name the "trimmed-tree" Knuth-Yao algorithm which we show provably supports random sample privacy.

To the best of our knowledge, the idea of using a modified Knuth-Yao algorithm for constant-time discrete Gaussian sampling was introduced by [KRR+18a] and [KRVV19a] is the only work investigating its performance. Building on their empirical insights, we offer non-trivial provable guarantees on the efficiency of trimmed-tree Knuth-Yao which also hold for random variables beyond the Gaussian. We find that while the entropy of the distribution is the key complexity parameter for standard Knuth-Yao, the finite bit precision b_p used to represent probability values is the analogous key complexity parameter for trimmed-tree Knuth-Yao. In particular, for sub-Gaussian variables with variance factor σ^2, we conclude that the trimmed-tree Knuth-Yao algorithm can be implemented as a circuit of size $O(\sigma b_t b_p^{3/2})$ where b_t is the number of bits for representing elements of the support and b_p is the number of bits for representing the probabilities of these elements. Furthermore, we prove the subtrees in the algorithm are small, proving that the the trimmed-tree Knuth-Yao can be computed by b_p circuits of size $O(\sigma b_t \sqrt{b_p})$ in parallel and then $O(b_t b_p)$ sequential operations on the outputs. This is an exponential improvement over the standard distribution-agnostic exponential bounds of $O(2^{b_p})$ or $O(2^{b_t} b_p)$. For certain parameter regimes we also achieve a lower bound on the overall circuit size of $\widetilde{\Omega}(b_t \cdot (\sigma + b_p))$, where here we suppress log factors. (Note that in the body, we not only expose log factors but also the constants behind our asymptotic results.) To achieve the more general theorems behind these provable guarantees, we carefully characterize how finite bit precision impacts sampling error for specific distributions based on their variance, higher-order moments, and a discrete analogue of Lipschitz-like smoothness conditions and its relation to differential privacy.

Along the way, we contribute to the literature offering possible solutions to the privacy vulnerabilities introduced by implementing differential privacy (DP) as a naïvely continuous mathematical object operating on discretized computers. As pointed out by [Mir12], "Unfortunately, simplicity of [a differentially private] procedure is deceptive, since once implemented in floating point arithmetic, its behavior deviates markedly from its mathematical abstraction" which may cause it to violate its purportedly provable privacy guarantees (pg. 5). To address this, we take seriously the title of [Mir12]'s work, "On Significance of the Least Significant Bits For Differential Privacy," both in our proof techniques for establishing our main theorems, and in applying our findings to DP settings such as calculating means for relatively large datasets while simultaneously maintaining privacy for individual data subjects. In particular, if t is the size of the support and n is the number of data subjects, then the trimmed-tree Knuth-Yao circuit size for sampling the noise for private mean estimation is bounded by $O((t/n) \cdot \sqrt{\log(n)} b_t b_p^{3/2})$[1] which again can be run more efficiently in parallel. Thus we prove that trimmed-tree Knuth-Yao performs increasingly well as the size of the dataset grows with respect to the size of the support, while otherwise losing no more than square root factors compared to linear dependence in the finite bit precision (which is removed when parallelized) and other relevant complexity parameters.

Finally, our results contribute to the literature which proposes bespoke variants of tree-based data structures in order to support cryptographic constructions. In particular, we use trimmed DDG trees as a primitive for constructing a random variable commitment scheme for the discrete Gaussian. To support this endeavor, we state our complexity results for the trimmed-tree Knuth-Yao algorithm in terms of Boolean circuit complexity, so that standard techniques for implementing Boolean circuits in zero knowledge (ZK) can be thoughtfully applied.[2] Our Boolean circuit implementing trimmed-tree Knuth-Yao takes as input a string of random bits with length equal to the finite bit precision. We can source each of these input random bits from the random bit commitment scheme of [BGKW24]. Our discrete Gaussian commitment scheme construction then implements the trimmed-tree Knuth-Yao algorithm on these hidden bit inputs within an arbitrary ZK proof system. By following the approach of [BGKW24], our discrete Guassian commitment scheme can then be used as a primitive for constructing certified probabilistic mechanisms, with applications such as certified differential privacy.

[1] This is true more generally, for mechanisms that can be made differentially private by sampling Gaussian noise with variance independent of domain size.

[2] For an in-depth discussion of the trade-offs between different ZK proof systems grounded in both underlying theory and concrete practice, based on considerations such as circuit depth and size, see e.g. [Tha22].

Our contributions Sampling from non-uniform randomness is increasingly important for cryptographic applications, such as lattice-based cryptography or differential privacy. Paramount to the guarantees of these mechanisms is keeping the internal randomness hidden. This presents technical challenges when we also want to certify the random mechanism was correctly run without breaking the mechanism's guarantees ("certified probabilistic mechanisms" [BGKW24]), or be resistant to timing attacks where variable runtime can leak the internal randomness used ([BBE+19b]). In this paper we present a provably efficient sampler that maintains random sample privacy, or *random sample hiding*, and is applicable to arbitrary discrete random variables. Specifically:

1. We present the "trimmed-tree" Knuth-Yao algorithm as a constant-time sampler (i.e., it does not have variable running time), allowing for RV commitment schemes and timing attack resistance. The core construction relies on the discrete distribution generating (DDG) tree, where leaves represent elements with specific bit values in their probability, trimmed to a fixed bit precision. This DDG tree can be constructed for any discrete distribution on finite support.[3] Computing the whole DDG tree produces a random sample from the distribution while maintaining random sample hiding.

2. We prove the Boolean circuit complexity of the trimmed-tree Knuth-Yao algorithm for sampling a σ^2-sub-Gaussian random variable is $\leq O(\sigma b_p^{3/2} b_t)$ where b_t is the bits for representing the domain, and b_p is the bits for precision of the PDF values, which is an exponential improvement over the naïve bounds, and avoids the exponential dependence on b_t of established constant-time samplers. Furthemore, we prove the DDG tree is made up b_p size-$O(\sigma\sqrt{b_p})$ subtrees, proving that trimmed-tree Knuth-Yao can be be computed by running b_p circuits of size-$O(\sigma\sqrt{b_p}b_t)$ and then $O(b_p b_t)$ sequential operations on the outputs. Further, this approach *uses only b_p random bits in a read-once manner*. Our proof follows from observing the sampling error introduced from finite-bit precision controls the size of the discrete distribution generating (DDG) tree, and then bounds this finite-bit precision error using the moments of the random variable.

3. We provide a lower-bound of $\Omega((\sigma - \log(\sigma) + b_p)b_t)$ on the circuit complexity of the trimmed-tree Knuth-Yao algorithm under certain parameter regimes. To do so we use relations between the number of leaves at each level of the DDG tree and a discrete variant on the Lipschitz parameter of the distribution.

4. We characterize the parameter regimes in which constant-time Knuth-Yao outperforms rejection sampling. Rejection sampling is the main other algorithmic approach in the literature that can be used to sample from arbitrary discrete distributions with random sample hiding while avoiding exponential dependence on b_t for certain parameter regimes (see e.g. [HPRR20, DFW22, SZJ+22, WYF+23]).[4] At a high level, while rejection sampling is most efficient for distributions which are close to uniform, Knuth-Yao is most efficient for tightly concentrated distributions, such as those used in the applications we consider next.

5. We apply these circuits for trimmed-tree Knuth-Yao to constructing random variable commitment schemes for arbitrary discrete distributions, giving exponential improvements in the number of random bits and circuit complexity used for certified differentially private means and counting queries over large datasets and domains.

1.1 Related Work

Discrete Gaussian Samplers for Cryptographic Applications. Sampling from a discrete Gaussian has become central to lattice based cryptography [DDLL13a, DCCM22]. However, implementations are prone to side-channel attacks which use the varying runtime of samplers to infer values [BBE+19b]. Similarly,

[3] Note that discretization and finite support are both necessary characteristics for representing a probability distribution on a finite computer.

[4] While e.g. [BBE+19b] uses constant-time cumulative distribution table (CDT) lookup, they simply pass over the entire table in order to do so, so the circuit complexity of this approach is the same across all distributions and is exponential in b_t. In particular, the size of the table is $\Theta(b_p \cdot 2^{b_t})$ since the table varies over all elements in the support where each entry in the table is a probability value expressed up to b_p, so the circuit complexity to output a b_t-bit element in the support is $\Theta(b_p b_t \cdot 2^{b_t})$. See Section 1.1 for further discussion of the landscape of previous constant-time samplers.

sampling from a discrete Gaussian is important for differential privacy [CKS20], and similar timing-attacks could be instantiated [BV17]. Further, timing-attack resistance is a prerequisite to directly using such samplers for certified probabilistic mechanisms [BGKW24]. Samplers which operate in constant time would directly mitigate these issues.

Previous Constant-Time Samplers. Towards constant-time samplers of discrete Gaussians, past work has relied on using a constant-time version of CDT [BBE+19b] or Knuth-Yao [KRR+18a, KRVV19a], or rejection sampling [DFW22] which can also be isochronously done over a base CDT sampler [HPRR20, SZJ+22]. We will later directly compare our proposed trimmed-tree Knuth Yao algorithm to rejection sampling. However, the runtimes for these past algorithms have all been linear in the size of the domain, i.e., exponential in its bit representation. Note that while—for any of these methods—we could naïvely remove this exponential dependence for reasonably concentrated distributions by assuming a desired total variation distance to the true distribution and then using a matching concentration bound to cut off portions of the support, in this work we seek to avoid this assumption, leaving open the option for other notions of distributional closeness to be utilized depending on the application. For instance, [BLR+18] argues that Rényi divergence is often a more appropriate measure for security proofs in lattice-based cryptography. Towards faster samplers, [KRVV19a] empirically noticed the binary formulas for Knuth-Yao subtree can sometimes be far smaller than expected by the exponential worst-case bound for some discrete Gaussians. In particular, the size of the subtrees were empirically bounded by a small constant and this led to further runtime optimizations. We prove the subtrees are small for *all* sub-Gaussian random variables (RV), which also leads to a bound on the full circuit size with linear dependence on the sub-Gaussian parameter. We note that previous methods which build on sampling from a fixed truncated discrete Gaussian [BBE+19b, HPRR20, SZJ+22] can use our results for the base sampler.

Previous Certified DP Approaches. Past approaches to providing zero-knowledge proofs of differential privacy (in general certified probabilistic mechanisms) focused on the particularly cryptographically convenient Binomial mechanism [BGKW24].[5] For this special case, they needed $O(2^{b_t})$ random bit commitments as one needs to sample a random bit for each element in the domain, which are then summed in an arithmetic circuit of logarithmic depth in its size, namely $O(b_t)$ depth. In our work, we only need b_p random bit commitments, independent of the domain size, which we then operate on in a Boolean circuit of $O(b_t b_p^{3/2})$ size and $O(b_p)$ depth. Note that [BGKW24] investigates both theoretically and empirically how generating the random bit commitments is the most costly online component in their constructions of certified DP, and the only component which cannot be made non-interactive. Our work makes exponential improvements to the required number of random bit commitments.

An alternative to these zero-knowledge-based constructions is utilizing secure multiparty computation (MPC) with dishonest parties, or more specifically, secure two-party computation (2PC) in the malicious security threat model. One recent work improves the efficiency of rejection sampling for discrete Gaussian distributions in the MPC setting [WYF+23]. Their solution takes a random bitstring as the MPC input from each party, then evaluates the XOR function on these inputs to produce a single trusted uniformly random bitstring, and finally composes the output of this function with a rejection-sampling-based discrete Gaussian sampler. Thus the MPC functionality being implemented is the composition of the XOR function and the rejection-sampling-based discrete Gaussian sampler [WYF+23] proposes a specific Boolean circuit to implement this functionality and suggests applying generic maliciously-secure 2PC to this circuit, but does not characterize the complexity overhead that this produces. The example approach they highlight would involve sending many (independently garbled) copies of the circuit, sampling a subset of these copies, and evaluating multiple of these. In our work, we not only bound the circuit size of our discrete sampler, but also the (small constant) overhead of implementing each gate of this circuit using "witness-indistinguishable proofs of knowledge." We thereby provide security against potentially dishonest parties without requiring multiple evaluations of the sampling circuit. [WYF+23] also focuses specifically on the discrete Gaussian distribution, while we provide a construction and complexity analysis which applies for arbitrary discrete

[5] Alternative approaches to certifying differential privacy in zero-knowledge are known to be less efficient than the previously mentioned Binomial mechanism approach [BC23, BGKW24].

distributions. Further, there are general drawbacks to utilizing 2PC in this setting, in particular the challenges to providing efficient *publicly verifiable* MPC, which is a crucial property in the certified DP setting. See Section 1.3.1 of [BGKW24] for further discussion of downsides of utilizing 2PC for certifying differential privacy.

Previous Work Addressing Computer System Non-Determinism Insecurities for DP. We note another line of work towards certifying differential privacy only checks the execution of an algorithm whose guarantees are for continuous distributions [STC+24] and not the finite precision implementation. Privacy discrepancies are commonly caused by not accounting for error due to finite numerical precision: "the reason for this exploitable vulnerability is that floating-point implementations of differentially private algorithms deviate in important respects from their mathematical abstractions" [Mir12]. Floating point non-determinism is also a primary issue in verifiable machine learning training and data use [JYCC+21, SAB24, TJSP22]. In general, improving software determinism for floating point representations has been a goal for improving the security of computer systems [HRS+17, BLP22]. We offer a specific tool for mitigating the privacy issues due to finite representations in DP, as well as other settings which use similar probability distributions, by explicitly studying discrete random variables with PDFs represented to a specified finite bit precision.

Additionally, algorithms for security-sensitive applications with potentially unbounded runtimes, such as rejection sampling for certified DP, can lead to security issues when implemented on commercially-available computer architecture due to hardware non-determinism. Attacks which leverage non-determinism in software execution include [LSG+20, KHF+20]. In particular, [BV17] advocates for determinism in differentially private algorithms citing the possibility of timing attacks, which [CKS20] also raises for the specific case of sampling the discrete Gaussian for DP [6]. We prove that trimmed-tree Knuth-Yao offers an efficient constant-time sampler for a wide class of practically relevant distributions. However, while maintaining constant time addresses timing attacks, our construction does not necessarily remove other hardware side-channels, such as power consumption [WPH+22].

Previous Tree-Based Data Structures for Cryptographic Constructions. There are a number of previous works which also propose tweaking tree-based data structures so they can be used to construct cryptographic objects. One such example is using Merkle trees [Mer87] for constructing zero-knowledge sets [MRK03], which has led to a long line of work; see for instance [CHL+05, CFM08, LY10]. Merkle trees are also used in public key infrastructure [MBB+15, HHK+21]. Additionally, oblivious trees, a variant of 2-3 trees, can be used to preserve the privacy of the sequence of operations applied to a database in order to construct incremental digital signatures [Mic97]. This sparked a line of work on history independent data structures [NT01, HHM+05, CSSV23]. Binary tree structures are also used for oblivious RAM [SDS+18]. In this paper, we propose using a trimmed variant of the DDG tree for constructing discrete sub-Gaussian commitment schemes, amongst other cryptographic objects.

1.2 Technical Overview

1.2.1 Structure of the Manuscript

Within the organizational structure of this paper, we establish our trimmed-tree Knuth-Yao circuit complexity upper bound as follows. In Section 3 we bound the sampling error introduced by finite-bit precision by the concentration of the random variable (Lemma 1.1). Then in Section 4.2 we show the circuit size of trimmed-tree Knuth-Yao is related to the rate of error introduced by finite-bit precision (Theorem 1.1), and provide our $O(\sigma b_t b_p^{3/2})$ bound (Corollary 1.1). We establish lower-bounds on the Knuth-Yao circuit complexity in Section 4.3 (Theorem 1.2). A comparison of the Knuth-Yao upper and lower-bounds to rejection sampling is presented in Section 5 (Corollary 1.3 and Corollary 1.4), where we note exponential improvements for distribution far from uniform. Lastly, in Section 6 we construct random variable commitment schemes using trimmed-tree Knuth-Yao circuits, and then in Section 7 apply Knuth-Yao to certifying differential privacy for private mean estimation over large datasets and domains (Theorem 1.4).

[6] We note [CKS20] cited terminating after some specified time-limit as a means of introducing runtime determinism, but this neglects the possibility of learning information from shorter runs. Other sources of non-determinism, such as varying operations between runs, could also lead to side-channel attacks.

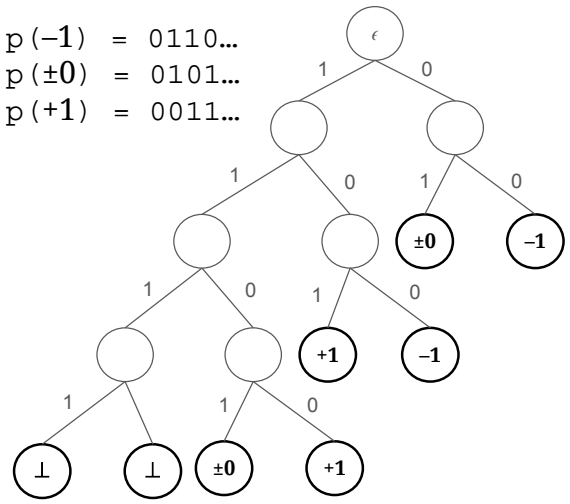


Figure 1: The Knuth-Yao DDG tree for a discrete distribution $\mathcal{D}$ over the finite support $[-1,1] := \{+1, \pm 0, -1\}$ (i.e. $b_t = 2$) trimmed to 4 bits of precision (i.e. $b_p = 4$). The root of the tree is indicated by the empty string ϵ and leaves are placed lexicographically based on the PDF values of elements in the support. For an element $x \in [-1,1]$, let $p_i(x)$ indicate whether this element's probability has a 1 at the ith bit position. If $p_i(x) = 1$, then in the DDG tree at depth i there exists a leaf labelled x. For example, since $p_3(-1) = 1$, at depth 3 there exists a leaf labelled -1. Further, if $p_i(x) = 1$, then there exists a random bitstring of length i which will lead Knuth-Yao to output x. For example, since $p_3(-1) = 1$, there exists a random bitstring of length 3 which will lead Knuth-Yao to output -1. Algorithmically, due to the lexicographic labelling of the tree's edges, one way that the Knuth-Yao algorithm could output -1 using the above DDG tree is if it randomly samples 100. We introduce the "error" token $\perp$ to absorb the leftover probability created by trimming the PDF values to finite bit precision.

1.2.2 Setting and Notation

Note that discretization and finite support are necessary requirements on any finite computer. Let $[-t,t] \cap \mathbb{Z}$ be the support of the desired discrete distribution $\mathcal{D}$, which we will denote simply by $[-t,t]$ throughout this overview since we only consider distributions with discrete support. Each element of the support can thus be represented using $b_t := \lceil \log(t+1) \rceil + 1$ bits. We will also need to approximate the PDF of $\mathcal{D}$ to a fixed bit precision so that each PDF value can be represented using b_p bits. We now describe our contribution towards constant-time samplers and hence samplers amenable to timing-attack resistance, RVCS, and thereby also the needs of accurate private probabilistic mechanisms.

1.2.3 The Knuth-Yao Algorithm

We now recall the construction of the Knuth-Yao algorithm, and a constant-time version. The Knuth-Yao algorithm is a standard method for sampling from discrete probability distributions with bounded support. It can be understood as a read-once branching program or binary decision diagram based on the discrete distribution generation (DDG) tree of $\mathcal{D}$ which takes as input a sequence of random bits, which determines how to traverse the tree, and ultimately outputs an element in the support, determined by which leaf in the tree has been reached according to this sequence. This algorithm is considered standard for discrete distributions because within an additive constant, the expected number of random bits utilized by Knuth-Yao is optimal. Specifically, if $H(\mathcal{D})$ is the entropy of distribution $\mathcal{D}$, Knuth-Yao needs no more than $H(\mathcal{D})+2$ random bits in expectation. See e.g. [Dev86] for a treatment of the standard Knuth-Yao algorithm's random bit complexity, as well as for contextualization of its relationship with non-discrete random sampling. In our case, we cannot immediately use the standard Knuth-Yao algorithm because it is known to not be timing-attack resistant. So, we also present a trimmed-tree variant on the Knuth-Yao algorithm which is timing

attack resistant, following similar construction in past work [KRR+18b, KRVV19b].

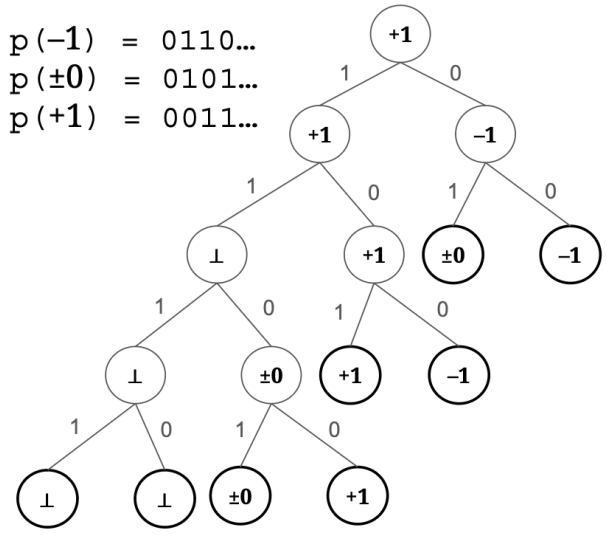


Figure 2: The Knuth-Yao DDG tree for a discrete distribution $\mathcal{D}$ over the finite support $[-1,1] := \{+1, \pm 0, -1\}$ (i.e. $b_t = 2$) trimmed to 4 bits of precision (i.e. $b_p = 4$), labelled according to the constant-time sampling algorithm for random b_p-bit input $r = 1011$. Each leaf is labelled as in Fig. 1 and each internal node is labelled to match the child indicated by r_i: letting c_1 and c_0 denote the left and right children of the node respectively, it is labelled the same as c_{r_i}. For example, since $r_0 = 1$, the root node is labelled to match its left child c_1 which has label $+1$. The sample that is output by this algorithm is the label of the root node, here $+1$.

Standard Knuth-Yao Algorithm. Formally, recall the Knuth-Yao algorithm is based on the *discrete distribution generation (DDG) tree* of $\mathcal{D}$. Let $p : [-t, t] \to [0, 1]$ be the PDF of $\mathcal{D}$. For each $x \in [-t, t]$ (or more generally the support), let $p_i(x)$ indicate the ith bit of $p(x)$. Consider a binary tree with lexicographically labeled nodes, e.g. nodes at depth i are labeled by strings in $\{0,1\}^i$ in the standard manner. Then depth i of the DDG tree is as follows: for each $x \in [-t, t]$ such that $p_i(x) = 1$, place a leaf at the lexicographically first available node and label that leaf with x.[7] Any remaining available nodes become internal nodes that generate two available nodes each for depth $i+1$. Continue as long as available nodes remain at depth $i+1$. The Knuth-Yao algorithm works by traversing the path through this tree that is indicated by the string of random bits it generates until it hits a leaf, in which case it outputs the label of that leaf. For instance, the random bits $\{r_0, r_1, \ldots, r_b\}$ will be the entirety of the random bits generated by the algorithm, causing it to terminate and output some $x \in [-t, t]$, if and only if node $r = r_0 r_1 \cdots r_b$ in the DDG is a leaf labeled x. Due to the structure of the tree and the fact that $\sum_{x \in [-t,t]} p(x) = 1$, there will always be an available node when the algorithm calls for one, and the tree will terminate with labeled leaves at depth d if and only if there exists x such that $p_d(x) = 1$ and there does *not* exist any x such that $p_i(x) = 1$ for $i > d$.

Trimmed-Tree Knuth Yao Algorithm. In order to make Knuth-Yao constant-time, we will sample from $\mathcal{D}_{b_p}$ instead of $\mathcal{D}$, where $\mathcal{D}_{b_p}$ is defined as the probabilities in $\mathcal{D}$ truncated at b_p bits, with remaining probability going to an error element $\perp$. For Knuth-Yao, this entails cutting off the DDG tree after depth b_p. The DDG for $\mathcal{D}_{b_p}$ is illustrated in Fig. 1. To make this now constant-time we will compute the whole tree to hide which leaf is returned; we immediately have the random bit complexity of trimmed-tree Knuth-Yao sampling from $\mathcal{D}_{b_p}$ is $b_{KY} = b_p$ (i.e., the maximum number of bits needed for a walk to any leaf). This computation over the entire DDG tree can be viewed as an OR over the random walks starting at each leaf of the reversed input bitstring. Specifically, to sample from $\mathcal{D}_{b_p}$ in constant time according to the trimmed-tree Knuth-Yao algorithm,

[7] The order in which the x's are considered can be arbitrarily chosen; for the purposes of this paper we choose lexicographic ordering for simplicity.

0. Let $r = \{r_0, r_1, \ldots, r_{b_p-1}\} \in \{0,1\}^{b_p}$ be the random input. Start at depth $i = b_p$.
1. Recall that each leaf in the DDG tree is labelled by an element in the support. For each internal node at depth i, label it the same as its corresponding child: letting c_1 and c_0 denote the left and right children of the node respectively, label the node the same as c_{r_i}. Repeat for $i - 1$.
2. Once the single root node at depth $i = 0$ is labelled, output the label of this root node.[8]

See Fig. 2 for an example of this sampling procedure.

1.2.4 Knuth-Yao Upper Bound Through Finite Precision Analysis

As we move from using standard Knuth-Yao to a constant-time variant of Knuth-Yao to draw from a distribution $\mathcal{D}$ (Section 4), the fixed bit precision b_p utilized to express this distribution's PDF becomes crucial. Immediately, note that in order to make Knuth-Yao constant-time, we cannot allow the DDG tree to have unbounded depth as it can in the standard case, and instead the tree must be trimmed at some fixed depth. In particular, due to the construction of the DDG tree, its depth corresponds to the maximum number of bits needed to express the PDF value of each element in its support. This entails an unbounded depth for distributions with irrational PDF values, such as the Gaussian. To avoid this, we can simply decide an appropriate bit precision for the PDF given the application and trim the DDG after depth b_p. For instance, in the lattice cryptography or certified probabilistic mechanism settings, b_p would be set based on the desired security parameter, while in the DP setting, b_p would be set based on the desired δ parameter. Now we immediately conclude that the random bit complexity of this trimmed-tree Knuth-Yao algorithm is no longer based on the entropy of the distribution but is simply b_p, suggesting that bit precision may be key to meaningfully characterizing this setting.

Building on this insight, in our work we show that carefully characterizing the sampling error *for a given distribution* $\mathcal{D}$ that is introduced by fixing the PDF bit precision to b_p simultaneously characterizes the size of the subtree and hence the overall size of the b_p-depth DDG tree for that specific distribution, and thereby the time complexity of trimmed-tree Knuth-Yao for $\mathcal{D}$. Intuitively, this is because the amount of probability mass left unexpressed after depth i determines how much the DDG tree needs to have already grown by depth i in order to be prepared to accommodate that probability mass at deeper levels. This can be formally expressed by considering trimmed-tree Knuth-Yao to not be sampling from $\mathcal{D}$ itself but instead from the closely-related distribution $\mathcal{D}_{b_p}$ which introduces an "error" token $\perp$ outside of the original domain where all leftover probability mass created by trimming the PDF to b_p bits is placed. We provide bounds on $\Pr_{b_p}[\perp]$ based on the b_pth moment of $\mathcal{D}$, and when this bound is instantiated for the practically significant case of sub-Gaussian random variables with variance factor σ^2 (Lemma 1.1), we prove that the size of the fixed-depth DDG tree and thereby time complexity of trimmed-tree Knuth-Yao is $O(\sigma b_t b_p^{3/2})$ which can be parallelized by running b_p circuits of size $O(\sigma b_t \sqrt{b_p})$ and then applying $O(b_t b_p)$ sequential operations on the output (Theorem 1.1 and Corollary 1.1).

Lemma 1.1 (Informal)**.** *If $\mathcal{D}$ is sub-Gaussian with variance factor σ^2 then $\Pr_{b_p}[\perp] = O\left(\sigma \cdot \sqrt{b_p} \cdot (\frac{1}{2^{b_p}})^{\frac{b_p}{b_p+1}}\right)$.*

Theorem 1.1 (Informal)**.** *If $\mathcal{D}$ is sub-Gaussian with variance factor σ^2, then the size of the subtree rooted at 1^i0 in the b_p-bit precision DDG tree is $s_i = O\left(\sigma \cdot \sqrt{b_p}\right)$ and the size of the overall tree is $s_{b_p} = O\left(\sigma \cdot b_p^{3/2}\right)$.*

Corollary 1.1 (Informal)**.** *If $\mathcal{D}$ is sub-Gaussian with variance factor σ^2, then the size of its b_p-trimmed-tree Knuth-Yao circuit is $c_{KY} = O\left(\sigma \cdot b_t \cdot b_p^{3/2}\right)$. Moreover, it can be computed by running b_p circuits of size $O\left(\sigma \cdot b_t \cdot \sqrt{b_p}\right)$ in parallel and then applying $O(b_t \cdot b_p)$ sequential operations on the output.*

Remark 1. *Lemma 1.1 may be of general interest for analyzing the distribution-specific error introduced by finite bit precision in randomness sampling more broadly. In particular, Theorem 1.1 utilizes this characterization to bound the distribution-specific size of bounded-depth DDG trees, which is of natural theoretical*

[8] In order to cryptographically carry out this algorithm, we will need to output an integer element of $[-t, t]$. In this case, the special error label $\perp$ can be mapped to an arbitrary fixed element of the support e.g. 0, so that whenever $\perp$ is the label of the root node, 0 is the output.

interest. Finally, Corollary 1.1 clarifies the implications of this bound for the distribution-specific runtime of the trimmed-tree variant of Knuth-Yao which is used in settings of practical interest.

1.2.5 Knuth-Yao Lower Bound Through a Lipschitz Condition

To develop a fuller picture of trimmed-tree Knuth-Yao, we next investigate distribution-specific lower bounds on the size of the DDG tree (Section 4.3), starting with the maximal probability obtained by an element in the support. Intuitively, when many of the distribution's PDF values are small and thus only generate leaves after some large depth d, the tree will be mostly full until that depth, and thus have exponential size with respect to d. Suppose that, in accordance with the distributions used in the practical settings we consider in this work, $\mathcal{D}$ has a symmetric, monotonically decreasing PDF p around its maximal value $p(0)$ (e.g., a Gaussian). Let d_{min} indicate the position of the most significant bit in $p(0)$ namely $d_{min} := \lceil \log(1/p(0)) \rceil$. Due to the structure of the DDG tree for $\mathcal{D}$, it will be full up until d_{min}, so it follows that the size of the DDG tree is at least $2^{d_{min}+1} - 1 = \Omega(1/p(0))$. However, this does not account for the contributions of the most significant bits of the other elements of the support, beyond 0, which fall below this depth, and does not surface the concurrent dependence of the tree size on b_p.

We may hope to surface a dependence on the bit precision b_p by accounting for the significant bits from elements beyond $p(0)$, which intuitively constitute more of the tree size as σ becomes small and thereby the DDG tree becomes more stick-like. In this case, the layers below d_{min} through the maximum depth of b_p become increasingly important to characterizing the overall size of the tree. To address this, we consider additional structural properties of the distribution, namely a Lipschitz property on its PDF. Specifically, we consider discrete distributions which have a Lipschitz parameter L such that for all neighboring elements in the support, the difference in their PDF values is $\leq L$. Intuitively, the Lipschitz parameter L bounds how quickly the PDF can decrease away from the mean. Thus the Lipschitz property ultimately gives a lower bound on the depth at which each element contributes its first leaf to the DDG tree.

To get an overall bound, we start with the full tree up to d_{min} and then simply add up the first leaf contributions at each depth from d_{min} to b_p, which provides the lower bound stated in Theorem 1.2 and Corollary 1.2. When applied to the discrete Gaussian with $L \leq 2^{-b_p}$, this theorem results in a lower bound of $\widetilde{\Omega}(\sigma + b_p)$ when $2^{b_t} \geq \Omega(\sigma\sqrt{\ln(\sigma)})$ [9], in contrast to our upper bound of $O(\sigma \cdot b_p^{3/2})$ when $\sigma\sqrt{b_p} \geq \Omega(1)$.

Theorem 1.2 (Informal)**.** *For a class of distributions $\mathcal{D}$, with L-Lipschitz PDF p s.t $L \leq 2^{-b_p}$, the size of their b_p-bit precision DDG tree is $s_{b_p} \geq \Omega\left(b_p + \frac{1}{p(0)} - \log(1/p(0))\right)$.*

Corollary 1.2 (Informal)**.** *For a class of distributions $\mathcal{D}$, with L-Lipschitz PDF p s.t $L \leq 2^{-b_p}$, the size of their b_p-trimmed-tree Knuth-Yao circuit is $c_{KY} \geq \Omega\left(b_t \cdot \left(b_p + \frac{1}{p(0)} - \log(1/p(0))\right)\right)$ with total random bit complexity b_p.*

Remark 2. *While our theorem in Section 4.3 can handle parameter regimes beyond $L \leq 2^{-b_p}$, it is helpful to consider the $L \leq 2^{-b_p}$ condition as providing information on how to set b_p to match the specific distribution. Intuitively, $L \leq 2^{-b_p}$ for a tight L-Lipschitz parameter means that the bit precision is no larger than is needed to distinguish between PDF values for neighboring elements in the support. Namely, the discretization of the PDF for $\mathcal{D}_{b_p}$ is no more than what is sufficient to reflect differences in probabilities between elements for $\mathcal{D}$—or at least in their highest-order bits. We hypothesize that our lower bound could be strengthened to asymptotically match the upper bound by analyzing the contributions of the lower-order bits of all of the elements in the support, instead of only their most-significant bits like we do in this work.*

1.2.6 Rejection Sampling Comparison

The bounds we establish in this work are sufficient to help us meaningfully characterize the parameter regimes in which trimmed-tree Knuth-Yao provides a timing-attack resistant discrete Gaussian sampler that outperforms rejection sampling (Section 5). Note that while rejection sampling is not constant-time, it

[9] In this case, the normalizing constant of the truncated Gaussian is greater than the normalizing constant over $\mathbb{Z}$ minus a constant, giving $p(0) \geq \Omega(\sigma)$.

is resistant to timing attacks because after each previous failed draw the slate of the algorithm is wiped clean, so no information about the sampled element (other than its desired distribution) is revealed by the process of repetition. However, in practice, its potentially unbounded runtime can still lead to issues when implemented on commercially-available computer architecture due to software non-determinism, as we recall in Section 1.1.[10]

As we describe in Section 5, rejection sampling draws most easily from distributions which are close to uniform, and it is particularly expensive in its use of randomness. For e.g. the purposes of certified probabilistic mechanisms for differential privacy, distributions which are close to uniform are not typically very useful, and both circuit complexity and random bit complexity are key to concrete practicality. For example, Corollary 1.3 shows how the circuit size for trimmed tree Knuth-Yao is exponentially smaller than rejection sampling for $\sigma \leq O(1)$, and uses significantly fewer random bits.

Corollary 1.3. *If $\sigma \leq O(1)$, then $c_R \geq \Omega(2^{b_t} b_p)$ and $c_{KY} \leq O(b_t b_p^{3/2})$. Note rejection sampling uses at least $\Omega(2^{b_t} b_p)$ random bits, where trimmed tree Knut-Yao uses b_p.*

Note that in other settings such as for e.g. the uniform distribution, the story would be quite different. In that case $p(0) = \theta(2^{-b_t})$ so the number of iterations for rejection sampling is constant and computing the PDF no longer requires computing the irrational function exp, though random bit complexity considerations remain pertinent. A consequence of our Knuth-Yao lower-bound is that its circuit size will be exponentially worse than rejection sampling in this case (Corollary 1.4). We further discuss the considerations that are brought into play when cryptographically computing the PDF and cryptographically sourcing the random bits in Section 1.2.7. We also elaborate further on the implications of our results to differentially private mean estimation in Section 1.2.8.

Corollary 1.4. *If $p(0) \leq O(2^{-b_t})$ and the distribution is a truncated discrete Gaussian, letting b_p be fixed, we have $c_{KY} \geq \Omega(2^{b_t})$. In contrast, $c_R \leq O(b_t)$.*

1.2.7 Constructing Random Variable Commitment Schemes and Certified Private Probabilistic Mechanisms

The characterization of trimmed-tree Knuth-Yao we offer provides a crucial ingredient for constructing RV commitment schemes for discrete distributions on finite support, with particularly strong concrete efficiency when we specialize this procedure to the truncated discrete Gaussian. Random variable commitment schemes (or RVCS) were introduced by [BGKW24] and we recall them in Section 2.5.

Significant care must be applied to analyze how the error probability produced by the misbehavior of each party, especially of the Dealer, may perturb the desired discrete distribution $\mathcal{D}$. See [BGKW24] for an in-depth treatment of the particular guarantees associated with RV commitment schemes.

The RVCS construction we present in Section 6 gives us the following.

Theorem 1.3. *If $\mathcal{D}$ is discrete on bounded support, and there exists a random bit commitment scheme, then there exists a RV commitment scheme for $\mathcal{D}_{b_p}$.*

Our construction takes as input the random bit commitments of [BGKW24] and applies a "witness-indistinguishable proof of knowledge" (see Section 2.5) to execute the b_p-trimmed-tree Knuth-Yao boolean circuit on these input random bit commitments. This results in Knuth-Yao's output leaf label hidden within a commitment. Thereby we obtain random sample hiding and overall a sub-Gaussian commitment scheme.

Corollary 1.5. *On σ^2-sub-Gaussian random variables, our RV commitment scheme construction uses b_p random bit commitments and performs $O(c_{KY}) = O(\sigma \cdot b_t \cdot b_p^{3/2})$ work on these, only reading each bit commitment once in the algorithmic execution path, to output a random-sample-hiding draw from $\mathcal{D}_{b_p}$ on $[-t, t]$.*

Remark 3. *Note that our RVCS construction applies more broadly to truncated discrete distributions with discretized probabilities. In general, to be used for a cryptographic application, such a RV commitment will*

[10] Note if we have a maximum runtime, then random executions of rejection sampling that pass the time-limit introduce error into the distribution we are sampling from, which would need to be accounted for.

ultimately be opened by the Dealer at some point, but perhaps only after post-processing with other hidden values or to a broader audience beyond only the Player[11]*. We note that Section 4.3 of [BGKW24] shows how to use an RV commitment scheme for $\mathcal{D}_{b_p}$ for an appropriate choice of b_p as a tool to construct certified probabilistic mechanisms which are additive in $\mathcal{D}$. In Section 4.3 [BGKW24] proves that this results in certified DP if done for a distribution $\mathcal{D}$ which gives a differentially private additive noise mechanism. We establish that is indeed the case for appropriately truncated discrete distributions with discretized probabilities in the following section.*

1.2.8 Application to Differentially Private Mean Estimation

Lastly, we instantiate the comparison between Knuth-Yao and rejection sampling with a case study of private mean estimation with the Gaussian mechanism (Section 7). The primary observation is that the variance σ^2 for the Gaussian we sample from to get (ϵ, δ)-DP is proportional to the ratio of domain size to the dataset size.

So as the dataset size grows, and hence we can have more accurate private mean estimation (due to smaller variance), we have Knuth-Yao becomes exponentially faster to sample the noise needed than rejection sampling (Theorem 1.4). This is as eventually $\sigma \leq O(1)$, at which point our previous comparison between rejection sampling and trimmed-tree Knuth Yao applies. A characterization for arbitrary dataset sizes, with constants and $\sqrt{\ln()}$ terms is provided in Section 7.

Theorem 1.4. *For private mean estimation of elements in a fixed $[-t, t]$ with a dataset of size $n = O(t)$ using the truncated discrete Gaussian mechanism, the circuit complexity of rejection sampling the random variable is $c_R \geq \tilde{\Omega}\left(2^{b_t} \cdot (b_p + b_t + c_p)\right)$ where b_p is the bits of precision for probabilities, $b_t = \lceil log(t+1) \rceil + 1$, and c_p is the circuit complexity of computing the probability. In contrast, trimmed-tree Knuth-Yao is $c_{KY} \leq \widetilde{O}(b_t \cdot b_p^{3/2})$.*

[11] See Section 4.4 of [BGKW24] for futher discussion of the latter possibility.

Contents

1 Introduction **1**
- 1.1 Related Work . . . 3
- 1.2 Technical Overview . . . 5
  - 1.2.1 Structure of the Manuscript . . . 5
  - 1.2.2 Setting and Notation . . . 6
  - 1.2.3 The Knuth-Yao Algorithm . . . 6
  - 1.2.4 Knuth-Yao Upper Bound Through Finite Precision Analysis . . . 8
  - 1.2.5 Knuth-Yao Lower Bound Through a Lipschitz Condition . . . 9
  - 1.2.6 Rejection Sampling Comparison . . . 9
  - 1.2.7 Constructing Random Variable Commitment Schemes and Certified Private Probabilistic Mechanisms . . . 10
  - 1.2.8 Application to Differentially Private Mean Estimation . . . 11

2 Preliminaries **13**
- 2.1 Probability Distributions and Notation . . . 13
- 2.2 Concentration Bounds . . . 13
- 2.3 Differential Privacy . . . 14
- 2.4 Derivative Bounds, Lipschitz Conditions, and the Relation to Differential Privacy . . . 14
- 2.5 Random Variable Commitment Schemes and Witness Indistinguishable Proofs of Knowledge 15

3 Finite Bit Precision **15**
- 3.1 Upper Bound . . . 16

4 Trimmed-Tree Knuth-Yao Sampler **16**
- 4.1 Upper-Bound by Domain Size . . . 17
- 4.2 Parallelized Upper Bound by Subtree Size . . . 17
- 4.3 Lower Bound . . . 19

5 Rejection Sampling **21**
- 5.1 Circuit Size of Calculating the Discrete Gaussian PDF . . . 23

6 From Sampler to Random Variable Commitment Scheme **23**

7 Knuth-Yao is Better For Private Mean Estimation Over Large Datasets **24**

A Appendix: From Variance Factor to Lipschitz from Section 4.3 **29**

B Appendix: Proof of Corollary 4.3 **29**

2 Preliminaries

2.1 Probability Distributions and Notation

In this work, we consider discrete probability distributions which operate over a finite support that can be mapped[12] to the integers centered on 0.

Notation 1 (Discrete Finite Probability Distribution)**.** *Let $\mathcal{D}$ be a* discrete probability distribution over finite support $[-t,t] \cap \mathbb{Z}$ *with PDF* $p : [-t,t] \cap \mathbb{Z} \to [0,1]$. *Since we only consider discrete probability distributions in this work, throughout the paper we simply use $[-t,t]$ to denote $[-t,t] \cap \mathbb{Z}$.*

More technically, we will view discrete probability distributions as members of a discrete distribution family *parameterized by e.g. the number of bits to represent the support $b_t := \lceil \log(t+1) \rceil + 1$, so that $\mathcal{D}$ is shorthand for the distribution family $\{\mathcal{D}_{b_t}\}_{b_t \in \mathbb{N}}$. Other distributional parameters which we will consider in this work include the PDF bit precision b_p and the variance (factor) σ^2. Throughout this paper, results referencing distributions that are stated asymptotically are asymptotic over the discrete series of members of the distribution family defined by the relevant distributional parameters.*

While we will study general discrete finite distributions, we will in particular use the (truncated) discrete Gaussian as an example throughout the paper.

Definition 1 (Discrete Gaussian Distribution)**.** *We define the discrete Gaussian $N_{\mathbb{Z}}(\mu,\sigma)$ as the distribution with density function $p(x) \propto e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ on $\mathbb{Z}$. Similarly, we define the b_t-truncated discrete Gaussian $N_{\mathbb{Z}_{2t+1}}(\mu,\sigma)$ as the distribution with density function $p(x) \propto e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ on $\mathbb{Z}_{2t+1} = [-t,t]$ (here $[-t,t]$ is the representation for the operations over $\mathbb{R}$). In both cases we assume x, μ are in the domain, and addition follows the arithmetic for those groups (where as non-linear operations are done through $\mathbb{R}$).*

Note for many of the analyses involving b_t-truncated discrete Gaussians, we need to also have bounds on its normalizing constants. To do so we often adapt Fact 19 of [CKS20] for the b_t-truncated discrte Gaussian by using tail bounds.

Definition 2 (Total Variation Distance)**.** *Probability distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ have* total variation distance $TV(\mathcal{D}_1, \mathcal{D}_2) = \frac{1}{2}\sum_{x \in [-t,t]} |Pr_{\mathcal{D}_1}[x] - Pr_{\mathcal{D}_2}[x]|$.

2.2 Concentration Bounds

Our analysis of trimmed-tree Knuth Yao will leverage the concentration of the distribution we are sampling. We now recall inequalities and definitions related to the concentration of random variables.

Lemma 2.1 (Higher-Moment Markov)**.** *Let X be a random variable with mean 0 and $M_n := \mathbb{E}[|X|^n]$ be its nth moments. Then* $\Pr[|X| \geq a] \leq \frac{M_n}{a^n}$.

Definition 3 (Sub-Gaussian Distribution)**.** *A random variable X with mean 0 is* sub-Gaussian *with variance factor $\sigma^2 > 0$ (which we will call σ^2 sub-Gaussian) if* $\Pr[|X| \geq a] \leq 2\exp\left\{-\frac{a^2}{2\sigma^2}\right\}$.

Fact 1 (Discrete (Truncated) Gaussians are Sub-Gaussian)**.** *The discrete Gaussian $N_{\mathbb{Z}}(0,\sigma)$ and the truncated discrete Gaussian $N_{\mathbb{Z}_m}(0,\sigma)$ (for any m) are σ^2 Sub-Gaussian.*

Proof. See Corollary 17 of [CKS20] for the discrete Gaussian. The proof for truncated follows from noting the tail probability for the truncated Gaussian must be less than the discrete Gaussian given the probability around 0 is greater; $\int_{-t}^{t} dp_{N_{\mathbb{Z}_m}}(x) \geq \int_{-t}^{t} dp_{N_{\mathbb{Z}}}(x)$ as the normalizing constant for the truncated discrete Gaussian is smaller. □

Corollary 2.1 (Sub-Gaussian Moments)**.** *If a random variable X with mean 0 is sub-Gaussian with variance factor σ^2, then $Var[X] \leq \sigma^2$ and in general it has moments*

$$M_n := \mathbb{E}[|X|^n] \leq e^{n/e}\sigma^n n^{n/2}$$

[12] In particular, our complexity theoretic results assume this operation can be done in $O(1)$ time in our complexity parameters.

for $n \geq 2$ and for $n = 1$, $\mathbb{E}[|X|] \leq \sigma\sqrt{2\pi}$.

Fact 2 (All Bounded Random Variables are sub-Gaussian)**.** *Random vairables in $[-t, t]$ are at least t^2 Sub-Gaussian.*

2.3 Differential Privacy

We now recall the definition of differential privacy. To do so, we recall that we say a pair of datasets D, D' are neighbouring if they differ in one datapoint. Specifically, we consider replacement to define neighbours[13]:

Definition 4 (Neighbouring Datasets)**.** *Two datasets D, D' are* neighbours *if $D' = (D \setminus x_{old}) \cup x_{new}$ for some $x_{old} \in D$ and $x_{new} \in D'$*

Differential privacy asks that the outputs of a randomized mechanism does not change significantly between using neighbouring datasets.

Definition 5 (Differential Privacy)**.** *A randomized mechanism $M : \mathcal{D} \to \mathcal{O}$ over datasets of size n is* (ϵ, δ)-differentially private (DP) *if for any neighbouring datasets D, D' we have for all events $E \subset \mathcal{O}$:*

$$\Pr[M(D) \in E] \leq e^{\epsilon} \Pr[M(D') \in E] + \delta.$$

Note [CKS20] recently provided analysis for the discrete Gaussian over $\mathbb{Z}$, but their approach does not immediately apply once the domain is truncated[14]. Nevertheless, following standard analysis of the Gaussian over $\mathbb{R}$, one can show that carefully adding truncated discrete Gaussian noise to an operation over datasets provides (ϵ, δ)-DP.

Lemma 2.2 (The Truncated Discrete Gaussian Mechanism for DP)**.** *Let $f : \mathcal{D} \to \mathbb{Z}_{2t+1}$ be some deterministic map from datasets to finite elements $\mathbb{Z}_{2t+1}$. The univariate b_t-truncated discrete Gaussian mechanism $f(D) + N_{\mathbb{Z}_{2t+1}}(0, \sigma^2)$, for any t and sensitivity $\Delta = \max_{\text{neighbouring } D,D'} \|f(D) - f(D')\|_2$, satisfies (ϵ, δ)-DP when $\sigma^2 = \frac{2\Delta^2 \ln(2/\delta)}{\epsilon^2}$ for $\epsilon \leq 1$.*

Proof sketch. This follows from the calculations from Theorem 2.1 in [SU], noting the only requirements for the proof was the σ^2 Sub-Gaussian tail bound, and the density function being proportional to the Gaussian density formula, both of which are true for the truncated discrete Gaussian by Definition 1 and Fact 1. □

2.4 Derivative Bounds, Lipschitz Conditions, and the Relation to Differential Privacy

Definition 6 (Discrete Lipschitz)**.** *Throughout the paper we will define a "derivative" over $\mathbb{Z}$ as the increase from adding 1: $p'(x) = p(x+1) - p(x)$. We hence say p is L-*Lipschitz *if $|p'(x)| \leq L \ \forall x$.*

Finally, we note that if adding a random variable provides (ϵ, δ) differential privacy, then this implies derivative bounds on its distribution. Combining this with the DP guarantees of the b_t-truncated discrete (Lemma 2.2) immediately implies Lipschitz bounds, which we later use to reason about trimmed-tree Knuth Yao lower-bounds.

Lemma 2.3 (Additive DP Mechanisms Have Lipschitz PDFs)**.** *Suppose the mechanism $f(x) + y$ where $y \sim p(y)$ is (ϵ, δ)-DP for queries $f(x)$ which are 1-sensitive. We then have $p'(y) \leq (e^{\epsilon} - 1)p(y) + \delta$, resulting in parameter $L \leq (e^{\epsilon} - 1)p(0) + \delta$ for distributions symmetric around zero.*

Proof. By assumption we have $\forall y$ that $p(y+1) \leq e^{\epsilon} p(y) + \delta$. We can then conclude $p'(y) = p(y+1) - p(y) \leq (e^{\epsilon} - 1)p(y) + \delta$. □

[13] The notion of neighbours is often related to the mechanism we are making private. For mean estimation (the example studied in this paper) having the same dataset size is used to have the sensitivity of the operation decrease with the dataset size.

[14] For example, Lemma 6 of [CKS20] uses the Poisson summation formula which is for sums over all $\mathbb{Z}$

2.5 Random Variable Commitment Schemes and Witness Indistinguishable Proofs of Knowledge

Definition 7 (Informal, see [BGKW24])**.** *A* random variable commitment scheme (RVCS) *for distribution $\mathcal{D}$ consists of a commitment scheme CS, an honest Dealer, and an honest Player such that the following holds:*

0. *Correctness: when the honest Dealer and the honest Verifier interact, they produce a commitment to a random variable distributed as $\mathcal{D}$.*
1. *Cheating-Player Distributional Soundness: if the honest Dealer interacts with an arbitrarily adversarial Player, they produce a commitment to a random variable distributed as $\mathcal{D}$ modulo aborting to output $\perp$.*
2. *Cheating-Dealer Distributional Soundness: if the honest Dealer interacts with an arbitrarily adversarial Player, they produce an openable[15] commitment to a random variable distributed $\Pr[\perp]$-close to $\mathcal{D}$ in total variation distance.*

Definition 8 (Informal, see e.g. [Tha22].)**.** *An interactive protocol between two parties is a* proof of knowledge (PoK) *for a relation R mapping $(input, witness)$ pairs to $\{true, false\}$ if it satisfies (1) completeness between an efficient Prover and an efficient Verifier and (2) knowledge soundness, i.e. there exists a Knowledge Extractor which can efficiently compute the honest witness from rewound transcripts with high probability when given rewindable oracle access to any non-negligibly successful Prover.*

Definition 9 ([FS90])**.** *A proof of knowledge is* perfect witness indistinguishable *if the distributions over transcripts produced when the Prover is given distinct honest witnesses for the same input are identical, even when interacting with an arbitrarily adversarial Verifier.*

Proposition 2.1 ([FS90, CDS94, Mau09])**.** *Perfectly witness-indistinguishable proofs of knowledge for computing Boolean circuits on additively homomorphic commitments to bits can be implemented via Σ-protocols with $O(1)$ overhead per gate.*

This proposition is laid out in more detail in [BGKW24]. Note that *additively homomorphic* commitments to bits can be produced by the Pedersen commitment scheme construction based on the discrete log assumption [Ped92, CM99], but see also e.g. [Tha22] to open up many constructive possibilities with different concrete tradeoffs while sustaining clarifying theoretical grounding.

3 Finite Bit Precision

A consequence of finite precision is that this introduces error in sampling from arbitrary probability densities (due to rounding the density values). We now elaborate on the statistical error this introduces. We then describe how this error can be more sharply bounded by how concentrated the random variable is. In Section 4 we make a connection between the error introduced by finite-bit samplers and the circuit size of Knuth-Yao. In Section 7 we discuss how this sampling error affects differentially private guarantees.

Let $\Pr_{b_p}[\perp]$ indicate the leftover probability mass that results from truncating the PDF of $\mathcal{D}$ after b_p bits. We can conceptualize this as the probability of outputting an "error" token $\perp$. More formally,

Notation 2. *Let $\mathcal{D}$ be a probability distribution over finite domain $[-t, t]$ with PDF $p : [-t, t] \to [0, 1]$. Then let $\mathcal{D}_{b_p}$ indicate the probability distribution given by the PDF $p_{b_p} : [-t, t] \cup \{\perp\} \to [0, 1]$ defined by $p_{b_p}(x) = \lfloor p(x) \cdot 2^{b_p} \rfloor / 2^{b_p}$ for $x \in [-t, t]$ and $p_{b_p}(\perp) = \sum_{x \in [-t,t]} p(x) - p_{b_p}(x) =: \Pr_{b_p}[\perp]$.*

Fact 3. *The statistical distance between $\mathcal{D}$ and $\mathcal{D}_{b_p}$ is given by the probability of $\perp$, i.e. $TV(\mathcal{D}, \mathcal{D}_{b_p}) = \Pr_{b_p}[\perp]$.*

We will consider exact samplers for $\mathcal{D}_{b_p}$. To choose b_p appropriately, we will need to bound $\Pr_{b_p}[\perp]$ and thereby the statistical distance between our desired vs sampled distributions. Further, $\Pr_{b_p}[\perp]$ will play a crucial role in determining the efficiency of constant-time Knuth-Yao sampling.

[15] In a "proof of knowledge" sense, see the next definition.

A naïve bound is given by considering that for each element in the domain, $p(x) - p_{b_p}(x) \leq 2^{-b_p} =: \delta_p$, so overall $\Pr_{b_p}[\perp] \leq (2t+1)\delta_p$. However, for reasonably concentrated distributions we can do better, and avoid explicit dependence on t entirely.

3.1 Upper Bound

Notice that for elements of the domain such that $p(x) < \delta_p$, the bound $p(x) - p_{b_p}(x) \leq p(x)$ is tighter than $p(x) - p_{b_p}(x) \leq \delta_p$. We will optimize when we use each inequality by taking $p(x) - p_{b_p}(x) \leq p(x)$ for $x \geq a$ for some $a \in (0, t]$ to be chosen. This leads to the following theorem.

Theorem 3.1. *For a probability distribution over $[-t, t]$ with n^{th} moment M_n, we have the probability of error $\Pr_{b_p}[\perp]$ is bounded as follows:*

$$\Pr_{b_p}[\perp] \leq \min\left\{\frac{2t+1}{2^{b_p}}, \quad \left(\left(\frac{1}{2}\right)^{\frac{b_p}{n+1}} + 8M_n^{\frac{1}{n+1}}\right)\left(\frac{1}{2^{b_p}}\right)^{\frac{n}{n+1}}\right\}.$$

Proof. By using $p(x) - p_{b_p}(x) \leq \delta_p$ for $x \leq a$ and $p(x) - p_{b_p}(x) \leq p(x)$ else, we have from Lemma 2.1,

$$\Pr_{b_p}[\perp] \leq (2a+1)\delta_p + \Pr_{X \leftarrow \mathcal{D}}[|X| \geq a] \leq (2a+1)\delta_p + \frac{M_n}{a^n}.$$

Optimizing this bound based on the choice of a, we want $2\delta_p - n\frac{M_n}{a^{n+1}} = 0$ which gives $a := \left(\frac{nM_n}{2\delta_p}\right)^{\frac{1}{n+1}}$. The previous choice of a results in

$$\Pr_{b_p}[\perp] \leq \delta_p + 2^{1+\frac{n}{n+1}} \cdot n^{\frac{1}{n+1}} \cdot M_n^{\frac{1}{n+1}} \cdot \delta_p^{\frac{n}{n+1}} \leq (\delta_p^{\frac{1}{n+1}} + 8M_n^{\frac{1}{n+1}})\delta_p^{\frac{n}{n+1}}.$$

The proof now follows from taking the minimum of this and the bound $(2t+1)\delta_p$ coming from using $p(x) - p_{b_p}(x) \leq \delta_p$ for all x.

□

The moment bounds for sub-Gaussian (Corollary 2.1) gives the following.

Corollary 3.1. *If $\mathcal{D}$ is sub-Gaussian with variance factor σ^2 and $(e^{1/e}\sigma n^{1/2})^{\frac{n}{n+1}} \geq 2^{-\frac{b_p}{n+1}}$ for free parameter $n \geq 2$, then*

$$\Pr_{b_p}[\perp] \leq \min\{\frac{2t+1}{2^{b_p}}, \quad 9e^{1/e} \cdot \sigma \cdot \sqrt{n} \cdot (\frac{1}{2^{b_p}})^{\frac{n}{n+1}}\}.$$

Lower Bounds The above upper-bound on error relied on an upper-bound on the tail of the random variable. One could instead attempt to construct a lower-bound on error by using a lower-bound on the tail, e.g., from the mills ratio. However, doing this and ignoring errors present for $x \leq a$ was not enough for us to tightly analyze the truncated discrete Gaussian Knuth-Yao circuit size.

4 Trimmed-Tree Knuth-Yao Sampler

We now study the runtime of trimmed-tree Knuth-Yao, the constant-time version of the Knuth-Yao algorithm. Whereas before the runtime complexity was the same as the random bit complexity, i.e. the depth of the tree, in order to make the algorithm constant-time and hide which leaf (and thereby sampled label) was chosen, we will calculate the whole tree every time instead of only a single path through the tree, because the latter would reveal the depth of the leaf that the algorithm reached. In particular, we will consider the circuit size complexity of the sampler c_{KY} so that existing ZK proof system techniques can be applied to implement it as an RV commitment scheme. Generically, Knuth-Yao implies that each of the b_t output bits can be expressed as an b_p-variable Boolean formula, which for $b_p \geq 2$ could require a Boolean circuit of size

$> b_t \cdot \frac{2^{b_p}}{2b_p}$ [16]. Our question is, can we do better? We will do so by proving an upper bound on the size of the depth-b_p DDG tree for $\mathcal{D}_{b_p}$ which for reasonably concentrated distributions gives exponential improvement over the naïve tree size bound of $2^{b_p+1} - 1$.

4.1 Upper-Bound by Domain Size

For sub-Gaussian RVs, we have that the number of elements with probability $\geq 2^{-b_p}$ is small, leading immediately to a better bound on the size of the trimmed-tree Knuth-Yao tree s_{b_p}. Mapping from the leaves of the tree to the b_t output bits then introduces another factor of b_t for a total circuit size of the trimmed-tree Knuth-Yao algorithm $c_{KY} = b_t \cdot s_{b_p}$

Theorem 4.1. *If $\mathcal{D}$ is sub-Gaussian with variance factor σ^2 then $s_{b_p} = O(\sigma b_p^{3/2})$ and $c_{KY} = O(\sigma b_t b_p^{3/2})$.*

Proof. Note that by the tail bound for sub-Gaussian, for $|x| > 2\sqrt{\ln(2)}\sigma\sqrt{b_p + 1}$ we have $p(x) < 2^{-b_p}$. Thus only $|x| \leq \sqrt{\ln(2)}\sigma\sqrt{b_p + 1}$ are leaves in the b_p-bit precision DDG tree, giving a bound on the number of leaves at each level. Note the number of internal nodes on a level is less than the leaves on the subequent level, meaning each level has at most $O(\sigma \cdot \sqrt{b_p})$ nodes. Summing over the b_p levels gives the desired bound on tree size. □

Notice that for the sub-Gaussian case, we have made an exponential improvement over the naïve bound with respect to b_p. In general, as long as the b_p^{th} root of the b_p^{th} moment of $\mathcal{D}$ is subexponential in b_p, we get a meaningful improvement. Another immediate bound for the size of the DDG tree would be $O(t \cdot b_p)$, so we have also made an exponential improvement over a naïve bound with respect to b_t.

While this shows the trimmed KY tree can be computed exponentially faster for concentrated RVs, we show that in fact the trimmed KY has additional regularity; it is made up of several small sub-trees. This leads to much faster parallelized implementations.

4.2 Parallelized Upper Bound by Subtree Size

[KRVV19b] noticed empirically that when running constant-time Knuth-Yao for the discrete Gaussian distribution, after the first input random bit was a 0, the DDG tree would always terminate at a leaf within a small addition depth; this fact allows for various optimizations, and in this paper we emphasize its impact on efficient parallelization. Namely, let s_i indicate the size and d_i indicate the depth of the subtree rooted at 1^i0. They observed empirically that each d_i was small and grew modestly with σ^2. Then the size of the subtree is bounded by $s_i \leq 2^{d_i+1} - 1$. Altogether, the total size of the b_p-bit precision DDG tree is $s_{b_p} = b_p + \sum_{i=0}^{b_p-1} s_i \leq \sum_{i=0}^{b_p-1} 2^{d_i+1}$.

Recall, mapping from the leaves of the tree to the b_t output bits then introduces another factor of b_t for a total circuit size of $c_{KY} = b_t \cdot s_{b_p}$. [KRVV19b] leave bounding d_i and thereby the circuits of the subtrees and c_{KY} as an open problem.

To resolve this open problem, our insight is to use the relation between the DDG tree structure and the underlying probability distribution. In particular, we will leverage our characterization of the error introduced by finite bit precision in order to bound d_i. For our purposes, what is crucial to note about the DDG tree is the following:

1. As long as there is probability mass remaining, the leftmost branch of the tree (i.e. the 1^i branch) continues branching to the left.
2. Whenever a branch to the right is added (i.e. a 1^i0 appears), this subtree rooted at 1^i0 will be terminated as soon as an additional $1/2^{i+1}$ of probability mass is accumulated as we continue to

[16] This result can be proven via probabilistic argument for a randomly chosen Boolean function, but concretely specifiable such functions remain unknown so far. See e.g. [Pap94] for a classic presentation.

deeper levels of the tree.[17]

Altogether this implies that the depth d_i of this subtree will be equal to the number of additional levels needed to get from $p_i = \sum_{j=1}^{i} 1/2^j$ to $p_{i+1} = \sum_{j=1}^{i+1} 1/2^j$, or equivalently to get to $p_i^{\perp} = 1/2^{i+1}$ remaining probability mass (the probability of all the leaves from the branch to the left 1^{i+1} which is only started to be filled after the 1^i0 subtree is terminated by definition of the DDG tree). We know that the root of the subtree is at level $i+1$. Thus we are interested in the depth, i.e. minimum number of random bits b_i, required to get to $\Pr_{b_i}[\perp] \leq p_i^{\perp}$. Then $d_i = b_i - (i+1)$.

With this approach we get the following lemma for a bound on the size of trimmed-tree Knuth-Yao tree.

Lemma 4.1. *For a distribution $\mathcal{D}$ over $[-t, t]$, and M_n its n^{th} moment, we have the the size the ith subtree is bounded as follows:*

$$s_i \leq \left((\frac{1}{2^{b_p}})^{\frac{1}{n+1}} + 8M_n^{\frac{1}{n+1}} \right)^{(n+1)/n} 2^{(i+1)/n}.$$

Moreover, the size of the trimmed-tree Knuth-Yao tree of b_p precision is bounded as follows:

$$s_{b_p} \leq 4 \cdot ((\frac{1}{2})^{\frac{b_p}{b_p+1}} + 8M_{b_p}^{\frac{1}{b_p+1}})^{(b_p+1)/b_p} \cdot b_p.$$

Equivalently, letting $C_{n,b_p} = (\frac{1}{2^{b_p}})^{\frac{1}{n+1}} + 8M_n^{\frac{1}{n+1}}$ we have: $s_i \leq C_{n,b_p}^{(n+1)/n} 2^{(i+1)/n}$ $s_{b_p} \leq 4 \cdot C_{b_p,b_p}^{(b_p+1)/b_p} \cdot b_p$.

Proof. Using Theorem 3.1, we will indeed get the needed bound if

$$\frac{1}{2^{i+1}} \geq C_{n,b_p} \cdot (\frac{1}{2^{b_i}})^{\frac{n}{n+1}},$$

so $b_i = \frac{n+1}{n} \log\left(C_{n,b_p}\right) + \frac{n+1}{n}(i+1)$ suffices. Thus

$$d_i \leq b_i - (i+1) \leq \frac{n+1}{n} \log\left(C_{n,b_p}\right) + \frac{1}{n}(i+1).$$

Then the size of the overall tree is bounded by

$$\begin{aligned} s_{b_p} &\leq 2 \sum_{i=0}^{b_p-1} 2^{d_i} \leq 2 \cdot C_{n,b_p}^{(n+1)/n} \cdot \sum_{i=0}^{b_p-1} 2^{(i+1)/n} \\ &\leq 2 \cdot C_{n,b_p}^{(n+1)/n} \cdot 2^{1/n} \cdot \frac{2^{b_p/n} - 1}{2^{1/n} - 1} \leq 2 \cdot C_{n,b_p}^{(n+1)/n} \cdot \frac{2^{1/n}}{2^{1/n} - 1} \cdot \left(2^{b_p/n} - 1\right). \end{aligned}$$

Noticing that $\frac{2^{1/n}}{2^{1/n}-1} \leq 2n$ and setting $n = b_p$, we have $s_{b_p} \leq 4 \cdot C_{b_p,b_p}^{(b_p+1)/b_p} \cdot b_p$. This gives the theorem statement.

□

Bounding C_{b_p,b_p} gives the following theorem.

Theorem 4.2. *For the b_p-bit precision DDG tree for a distribution $\mathcal{D}$, we have $s_i \leq O\left(M_{b_p}^{1/b_p}\right)$ and $s_{b_p} = O\left(M_{b_p}^{1/b_p} \cdot b_p\right)$ if $\exists C' > 0$ such that $M_{b_p}^{\frac{1}{b_p+1}} \geq C'$ for $b_p \geq 1$, where M_{b_p} the b_p^{th} moment of $\mathcal{D}$.*

[17] This can be an established via a recursive argument which notes that half of the probability mass of $\mathcal{D}$ is sent along each edge when a given internal node of the tree branches in two.

Proof. Note $C_{b_p,b_p} = (\frac{1}{2}^{b_p/(b_p+1)} + 8M_{b_p}^{\frac{1}{b_p+1}}) \leq (\sqrt{2} + 8M_{b_p}^{\frac{1}{b_p+1}})$, so if $\exists C' > 0$ such that $M_{b_p}^{\frac{1}{b_p+1}} \geq C'$ then $\frac{\sqrt{2}}{C'} M_{b_p}^{\frac{1}{b_p+1}} \geq \frac{1}{2}^{b_p/(b_p+1)}$ and thus

$$s_{b_p} \leq 4(8 + \frac{\sqrt{2}}{C'})^{\frac{b_p+1}{b_p}} M_{b_p}^{\frac{1}{b_p}} b_p \leq 4(8 + \frac{\sqrt{2}}{C'})^2 M_{b_p}^{\frac{1}{b_p}} b_p$$

by Lemma 4.1. An analogous inequality follows for s_i. This gives the theorem statement. □

For sub-Gaussian random variables, we can get a different bound on C_{b_p,b_p}.

Theorem 4.3. *If $\mathcal{D}$ is sub-Gaussian with variance factor σ^2, $\sigma\sqrt{b_p} \geq e^{-1/e}$ and $b_p \geq 2$ then we have $s_i \leq O\left(\sigma \cdot \sqrt{b_p}\right)$ and the size of its b_p-bit precision DDG tree is $s_{b_p} = O\left(\sigma \cdot b_p^{3/2}\right)$.*

Proof. For sub-Gaussian with variance parameter σ^2 and moments ≥ 2 we have $C_{b_p,b_p} = (\frac{1}{2}^{b_p/(b_p+1)} + 8(e^{1/e}\sigma b_p^{1/2})^{\frac{b_p}{b_p+1}}) \leq (\sqrt{2}+8(e^{1/e}\sigma b_p^{1/2})^{\frac{b_p}{b_p+1}})$. Let $C' \leq \frac{1}{e^{1/e}}$, then we get $\sqrt{2} \leq (\frac{\sqrt{2}}{8(e^{1/e}C')^{\frac{b_p}{b_p+1}}})8(e^{1/e}\sigma b_p^{1/2})^{\frac{b_p}{b_p+1}} \leq \frac{\sqrt{2}}{8e^{1/e}C'} 8(e^{1/e}\sigma b_p^{1/2})^{\frac{b_p}{b_p+1}}$ where for the first inequality we used $C' \leq \sigma\sqrt{b_p}$ and for the second $C'e^{1/e} \leq 1$. So we conclude for $C' \leq e^{-1/e}$ and taking $\sigma\sqrt{b_p} \geq C'$ we get

$$s_{b_p} \leq 4(\frac{\sqrt{2}}{e^{1/e}C'})^{\frac{b_p+1}{b_p}} (e^{1/e}\sigma b_p^{1/2}) b_p \leq 4(\frac{\sqrt{2}}{e^{1/e}C'})^2 e^{1/e}\sigma b_p^{3/2}$$

where the last inequality used the fact $\frac{\sqrt{2}}{e^{1/e}C'} \geq \sqrt{2} \geq 1$ so larger powers lead to an upper-bound. We conclude the theorem statement by taking $C' = e^{-1/e}$. Analogous inequalities lead to the s_i bound. □

This approach gives a bound on the overall circuit size complexity for the trimmed-tree Knuth-Yao algorithm.

Corollary 4.1. *The overall size of the trimmed-tree Knuth-Yao circuit for distribution $\mathcal{D}_{b_p}$ is $c_{KY} = O(b_t \cdot M_{b_p}^{1/b_p} \cdot b_p)$ if $\exists C' > 0$ such that $M_{b_p}^{\frac{1}{b_p+1}} \geq C'$ for $b_p \geq 1$, where M_{b_p} is the b_p^{th} moment of $\mathcal{D}$. If $\mathcal{D}$ is sub-Gaussian with variance factor σ^2, then $c_{KY} = O(\sigma \cdot b_t \cdot b_p^{3/2})$ if $\sigma\sqrt{b_p} \geq e^{-1/e}$ and $b_p \geq 2$.*

Moreover we have the trimmed-tree Knuth-Yao algorithm can be more efficiently parallelized for sub-Gaussian RVs.

Corollary 4.2 (Parallelizability)**.** *If $\mathcal{D}$ is sub-Gaussian with variance factor σ^2, then computing the trimmed-tree Knuth-Yao circuit requires executing b_p size-$O(\sigma \cdot b_t \cdot \sqrt{b_p})$ circuits in parallel, and then performing $O(b_t b_p)$ sequential operations on the outputs of those circuits.*

Note the parallelized runtime is $O(\sigma \cdot b_t \cdot \sqrt{b_p} + b_t \cdot b_p)$, which is now linear in the b_p parameter, improving the $b_p^{3/2}$ dependence from the domain size approach in Section 4.1.[18]

4.3 Lower Bound

We can get a meaningful lower bound on the circuit complexity of Knuth-Yao by focusing on distributions $\mathcal{D}$ with PDFs p that are symmetric, maximal at $p(0)$, monotonically decreasing, and L-Lipschtiz (Section 2.4). This is the case for the truncated discrete Gaussian (Definition 1).

First, let's establish some notation. For the DDG tree, let t_i indicate the total number of nodes at depth i in the tree, n_i indicate the number of internal nodes, and ℓ_i indicate the number of leaves. Then as pointed out by [Dev86], we have the recurrence relation $t_{i+1} = n_{i+1} + \ell_{i+1} = 2n_i$ with the base case $t_0 = n_0 + \ell_0 = 1$.

[18] Note that standard rebalancing techniques for Boolean formulas due to [Spi71] can be utilized to decrease the depth of the final sequential operations from b_p to $O(\log(b_p))$, but this comes at the cost of a polynomial increase in the circuit size.

Similarly to their proof of Theorem 2.1, this implies that $t_{i+1} = \sum_{j=i}^{b_p} \frac{\ell_{j+1}}{2^{j-i}}$, where compared to [Dev86] we have concluded the sum at depth b_p by taking $n_{b_p+1} = 0$ instead of allowing potentially unbounded depth. This corresponds to sampling from $\mathcal{D}_{b_p}$ as done in the trimmed-tree Knuth-Yao algorithm to have constant runtime.

Hence, by lower-bounding ℓ_i we can lower-bound the size of the DDG tree for $\mathcal{D}_{b_p}$ from certain depths. We do so by inspecting the role of how quickly the distribution can change, described by the Lipschitz parameter.

Lemma 4.2. *Let $s_{d:b_p}$ denote the size of the DDG tree from depth d to b_p. If $\mathcal{D}$ has a symmetric, monotonically decreasing, L-Lipschitz PDF p which we denote as maximal at 0, then*

$$s_{d:b_p} \geq \frac{5}{2} \cdot \frac{1/L}{2^{b_p}} \cdot (b_p - d + 1)\,.$$

Proof. We can lower bound ℓ_i based on the PDF in the following manner. Since all x in the support with PDF values where the first significant figure is in the $(i+1)$th bit will certainly have leaves at depth $i+1$ in the DDG tree, let

$$P_{\geq i+1}^{<i} := \{x \in [-t, t] \mid \frac{1}{2^{i+1}} \leq p(x) < \frac{1}{2^i}\}.$$

Then $\ell_{i+1} \geq \left|P_{\geq i+1}^{<i}\right| = 2(x_{\max} - x_{\min})$ for $x_{\min} = \arg\min_{x \in P_{<i}^{\geq i+1}} |x|$ and $x_{\max} = \arg\max_{x \in P_{<i}^{\geq i+1}} |x|$. Now due to the L-Lipschitz property of p,

$$\begin{aligned} p(x_{\min}) - p(x_{\max}) &= \sum_{x=x_{\min}}^{x_{\max}-1} p(x+1) - p(x) \\ \frac{1}{2^{i+1}} &\leq L(x_{\max} - x_{\min}) \\ 2(x_{\max} - x_{\min}) &\geq \frac{1}{L2^i} \end{aligned}$$

so $\ell_{i+1} \geq \frac{1}{L2^i}$.

Intuitively, we are saying the number of leaves at depth i is at least how many elements it takes to increase the probability by $\frac{1}{2^i}$, which is lower-bounded dividing $\frac{1}{2^i}$ by our upper-bound on how much adding one element can increase the probability, $\frac{1}{2^i L}$. Plugging this into the previously mentioned recurrence relation above, we find

$$\begin{aligned} t_{i+1} = \sum_{j=i}^{b_p} \frac{\ell_{j+1}}{2^{j-i}} \geq \sum_{j=i}^{b_p} \frac{\frac{1}{L2^j}}{2^{j-i}} \geq \frac{2^i}{L} \sum_{j=i}^{b_p} \frac{1}{4^j} &\geq \frac{2^i}{3L}\left(\frac{1}{4^{i-1}} - \frac{1}{4_p^b}\right) \\ &\geq \frac{1}{3L}\left(\frac{1}{2^{i-2}} - \frac{1}{2^{2b_p - i}}\right). \end{aligned}$$

Notice that for $i + 1 = b_p$, $t_{b_p} \geq \frac{5}{2} \cdot \frac{1/L}{2^{b_p}}$. Thus we have

$$s_{d:b_p} = \sum_{b=d}^{b_p} t_b \geq \frac{5}{2} \sum_{b=d}^{b_p} \frac{1/L}{2^{b_p}} \geq \frac{5}{2} \frac{1/L}{2^{b_p}} (b_p - d + 1)\,.$$

□

By combining this bound with the immediate bound of $\frac{1}{p(0)}$ for the first $d = \lceil \log(p(0)) \rceil$ levels due to the DDG tree being full until the most significant bit of the highest probability element, we get the following theorem.

Theorem 4.4. *If $\mathcal{D}$ has a symmetric, monotonically decreasing, L-Lipschitz PDF p which we denote as maximal at 0, then the size of its b_p-bit precision DDG tree is*

$$s_{b_p} \geq \Omega\left(\frac{1}{p(0)} + \frac{1/L}{2^{b_p}} \cdot \Big(b_p + \log(p(0))\Big)\right).$$

Proof. Letting $s_{i:j}$ for $j \geq i$ be the size of DDG treee between depths i to j, we have $s_{b_p} = s_{0:d} + s_{d:b_p}$. Now if we take $d = \lceil \log(1/p(0)) \rceil$, we have the DDG is full until d as d is the first bit to have a leaf in $\mathcal{D}_{b_p}$. Thus $s_{0:d} \geq 2^{\lceil \log(1/p(0)) \rceil}$.

Now substituting $d = \lceil \log(1/p(0)) \rceil$ into Lemma 4.2, we get $s_{d:b_p} \geq \frac{5}{2} \cdot \frac{1/L}{2^{b_p}} \cdot \Big(b_p - \lceil \log(1/p(0)) \rceil + 1\Big)$. Altogether, we get

$$\begin{aligned} s_{b_p} &\geq 2^{\lceil \log(1/p(0)) \rceil} - 1 + \frac{5}{2} \cdot \frac{1/L}{2^{b_p}} \cdot \Big(b_p - \lceil \log(1/p(0)) \rceil + 1\Big) \\ &\geq \frac{1}{p(0)} - 1 + \frac{5}{2} \cdot \frac{1/L}{2^{b_p}} \cdot \Big(b_p - \log(1/p(0)) + 1\Big) \\ &= \Omega\left(\frac{1}{p(0)} + \frac{1/L}{2^{b_p}} \cdot \Big(b_p - \log(1/p(0))\Big)\right). \end{aligned}$$

□

Note that the $\frac{1/L}{2^{b_p}}$ factor in the lower bound is strictly greater than 1 in the Lipschitz regime $L \leq 2^{-b_p}$. In this case we can think of this lower bound as scaling linearly both with $1/p(0)$, i.e. σ for truncated discrete Gaussian distributions, and b_p. Having small Lipschitz is intuitive for distribution close to uniform, and a consequence of Lemma 2.3 and Lemma 2.2 for truncated discrete Gaussian distributions with large σ.

The below corollary explicitly states the dependence on σ when the distribution is a truncated discrete Gaussian. While the statement supports general Lipschitz parameters, see Remark 2 for intuition for setting b_p in relation to L based on the role of the fraction $(1/L) \cdot 2^{-b_p}$ in this bound.

Corollary 4.3. *For the b_t-truncated discrete Gaussian with variance $\sigma^2 \geq 1$, L-Lipschitz, on domain $[-t, t]$ with $t \geq \sigma\sqrt{\ln \sigma}$, the size of its b_p-bit precision DDG tree is*

$$s_{b_p} \geq \Omega\left(\sigma + \frac{1/L}{2^{b_p}} \cdot \Big(b_p - \log(\sigma)\Big)\right).$$

The proof is in Appendix B, and in Appendix A we show a sufficiently large σ will make $L \leq 2^{-b_p}$.

5 Rejection Sampling

We now recall the complexity of rejection sampling, to serve as a baseline algorithm whose computation time also does not reveal the value of the sampled random variable. While there are some bespoke adjustments for the case of the discrete Gaussian which we bring to bear, even with these accounted for, we determine that the expected number of iterations and either the circuit size required for each iteration or the number of random bits required overall scales with the total size of the support. On the other hand, the number of iterations scales inversely with the square root of the variance, so if σ is large compared to t, i.e. the distribution is closer to uniform, the expected number of iterations may be small. Even in this case however, the circuit size required for each iteration or the random bit complexity will still scale linearly with b_t as well as b_p. Thus for the specific practical settings we consider, our work provides theoretical evidence that trimmed-tree Knuth-Yao offers significant strengths when compared to rejection sampling.

Concretely, rejection sampling using the uniform distribution works as follows:

0. Suppose $p(0)$ is the maximum PDF of the target distribution $\mathcal{D}$. Let $M := (2t+1) \cdot p(0)$.

1. Draw b_t bits uniformly at random. Draw an additional bit to determine the sign. This represents the chosen value in the support $x \in [-t, t]$.
2. Draw b_p bits uniformly at random. These represent some real number $r \in (0, 1]$. Calculate the PDF $p(x)$ to b_p bits of precision. If
$$r \leq \frac{p(x)}{M/(2t+1)} = \frac{p(x)}{p(0)},$$
return x. Otherwise loop back to step #1.

Notice that each iteration of this algorithm calls for $b_t + b_p + 1$ random bits. So, to understand the random bit complexity of the algorithm, we just need to characterize the number of iterations. This comes from the bound on the PDF ratio between our target distribution, e.g. the truncated discrete Gaussian, versus the distribution we have, i.e. uniform. In particular,

Proposition 5.1 (See e.g. [DDLL13b] presenting the classic work [vN51])**.** *Let $\mathcal{D}$ be a probability distribution over finite domain $[-t, t]$ with PDF $p : [-t, t] \to [0, 1]$ such that $p(0)$ is the maximum PDF of $\mathcal{D}$. Let $M := (2t + 1) \cdot p(0)$. Then in expectation, rejection sampling will take M iterations.*

Corollary 5.1. *The expected random bit complexity of rejection sampling is $b_R = (b_t + b_p + 1) \cdot M = (b_t + b_p + 1) \cdot (2t + 1) \cdot p(0)$. Let c_p indicate the circuit size required to compute the PDF p. Then the circuit size required for each of the expected M iterations of rejection sampling is $c_R = O(b_t + b_p + c_p)$.*

Determining c_p takes some care. First note that once we've evaluated the PDF of a sample, all that the circuit needs to do is compare it to the b_p-bit sample from $[0, 1)$ for $O(b_p)$ work. Mainly then, we encounter the issue of computing the PDF for each randomly drawn element, which requires computing the exponential function on the drawn element. Note that two common approaches for calculating the discrete Gaussian PDF lead either to $c_P = O(\sqrt{t \cdot b_p})$ with no increase in the random bit complexity, or to $c_P = O(b_t \cdot b_p)$ with the random bit complexity increasing to $b_R = (b_t \cdot b_p + b_t + b_p + 1) \cdot M = (b_t \cdot b_p + b_t + b_p + 1) \cdot (2t+1) \cdot p(0)$. We reserve fuller elaborations on these approaches to Section 5.1.

Comparing the complexity of rejection sampling to the bounds we have given for trimmed-tree Knuth-Yao leads to the following corollary for the specific case of the truncated discrete Gaussian.

Corollary 5.2. *If $\mathcal{D}$ is a truncated discrete Gaussian on $[-t, t]$ with variance factor $\sigma^2 \geq ((2^{b_p+1}\sqrt{b_p + 2}\frac{\sqrt{\ln(2)e}}{\sqrt{\pi}})^{\frac{1}{2}} + 1)^2$,and $t \geq \sigma\sqrt{\ln \sigma}$, then since $p(0) = \Theta(1/\sigma)$, for $b_p \geq 2$ the size of its b_p-trimmed-tree Knuth-Yao circuit is*
$$O(b_t \cdot b_p^{3/2} \cdot \sigma) \geq c_{KY} \geq \Omega\left(b_t \cdot (b_p + \sigma - \log(\sigma))\right)$$
with total random bit complexity b_p, and the total size of its b_p-bit-precision rejection circuits is

- $c_R = \Theta(2^{b_t} \cdot (2^{b_t/2} \cdot b_p^{1/2} + b_p) \cdot (1/\sigma))$ *with total random bit complexity* $b_R = \Theta(2^{b_t} \cdot (b_t + b_p) \cdot (1/\sigma))$*, or*
- $c_R = \Theta\left(2^{b_t} \cdot (b_t \cdot b_p) \cdot (1/\sigma)\right)$ *with total random bit complexity* $b_R = \Theta\left(2^{b_t} \cdot (b_t \cdot b_p) \cdot (1/\sigma)\right)$.

Proof. An immediate consequence of the parameter regimes which allow applying the trimmed-tree Knuth-Yao upper and lower bounds (Corollary 4.1 and Corollary A.1), and rejection sampling complexity (Corollary 5.1)

□

We now highlight specific parameter considerations.

Comparison to Knuth-Yao dependence on $p(0)$. Rejection sampling is non-exponential in domain bits when $p(0) = O(1/t)$. Theorem 4.4 implies trimmed-tree Knuth-Yao is at least exponential in domain bits when $p(0) = O(1/t)$. So for distributions close to uniform (in TV distance given TV distance will be $\leq \max\{p(0) - 2^{-b_t}, 2^{-b_t}\}$), we have rejection sampling is more efficient than Knuth-Yao. Else, for $p(0) = O(1)$ and $\sigma = O(1)$, Knuth-Yao will be exponentially faster.

Comparison to Knuth-Yao dependence on σ for Gaussian. We showed the trimmed-tree Knuth-Yao sampler's circuit size is bounded linearly by how concentrated the distribution is, represented by σ. We remark that rejection sampling depends inversely on σ, at least for the b_t-truncated discrete Gaussian (i.e., supported on $[-t, t]$), as stated in Corollary 5.2. To elaborate, as a consequence of bounds on the discrete Gaussian normalizing constant from Fact 19 of [CKS20] which is greater than the b_t-truncated discrete Gaussian, we have $\frac{1}{\sqrt{2}\sigma+1} \leq p(0)$. So by Corollary 5.1 we have an inverse σ trend for rejection sampling.

5.1 Circuit Size of Calculating the Discrete Gaussian PDF

In order to consider the circuit complexity costs of rejection sampling outside of the number of random bits required, we need to specify how the exponential function will be computed on an arbitrary element of the support x. We therefore dig into numerical methods for computing $p(x)$ to b_p bits of precision for a b_t-truncated discrete Gaussian. There are proposed approaches to avoid this:

1. Approximate the exponential function with an appropriately chosen polynomial [BBE+19a]. While [BBE+19a] gives a procedure to search for a minimal-degree and minimal coefficient-size polynomial to approximate the exponential function within a certain precision, they do not give a characterization of these minimal degree and coefficient size parameters. [AA22] analyze the asymptotics of the polynomial degree $d_{t',\delta}(f)$ in order to get a δ-close approximation of f for all inputs in $[0, t']$. They find that for $f(x) = \exp\{-x\}$, $d_{t',\delta}(f) \in \Theta(\sqrt{t' \cdot \log(1/\delta)}) = \Theta(\sqrt{t \cdot b_p})$ in our notation.[19] Evaluating this $d_{t',\delta}$-degree polynomial on a given input scales linearly with the degree as $\Theta(\sqrt{t \cdot b_p})$.
2. Apply rejection based on each bit of the random sample independently by utilizing precomputable thresholds (this utilizes the homomorphic property of the exponential function) [DDLL13b]. This involves increasing the number of bits sampled by another additive factor of $b_t \cdot b_p$ with $\Theta(b_t \cdot b_p)$ post-processing computation outside of this for each iteration. In settings where avoiding scaling linearly with the domain size t is important, this option becomes especially attractive since it only scales logarithmically with domain size as b_t, but it pays by increasing the number of random bits required by an additional linear additive factor in $b_t \cdot b_p$.

Proposition 5.2. *If $\mathcal{D}_{b_p}$ is a b_p-PDF-bit-precision b_t-truncated discrete Gaussian distribution (i.e., supported on $[-t, t]$), then*

1. *[BBE+19a]'s algorithm with [AA22]'s complexity analysis shows how to obtain either $c_p = \Theta(\sqrt{t \cdot b_p})$ with per-iteration random bit complexity $c_r = \Theta(b_t + b_p)$, and*
2. *[DDLL13b] shows how to obtain $c_p = \Theta(b_t \cdot b_p)$ with per-iteration random bit complexity $c_r = \Theta(b_t \cdot b_p)$.*

6 From Sampler to Random Variable Commitment Scheme

We now turn to the implications for RV commitment schemes (RVCS) from improved constant-time sampling via trimmed-tree Knuth-Yao. Recall that (informally) a random variable commitment scheme gives two parties a tool to perform the following interaction:

0. Both parties agree on a desired discrete distribution $\mathcal{D}$. One party takes on the role of the "Dealer" and the other takes on the role of the "Player."[20]
1. The two parties interact, resulting in the Dealer proposing a "commitment," which the Dealer claims holds a hidden random variable, to the Player.
2. The Player analyzes the transcript of their interaction with the Dealer, in hopes of determining whether

[19] Note that we are suppressing some subtleties in the Θ expression which do not impact our particular parameter regimes, but should be carefully checked by those utilizing the above complexity bounds for proposed cryptographic constructions.

[20] Note that in this preprocessing step, some additional housekeeping must be carefully done to set the stage for the cryptographic objects the parties use later in the interaction.

the Dealer has cheated with regards to their agreed-upon distribution, and either accepts the commitment or rejects it by sending the error token $\perp$.

3. As long as neither party has sent $\perp$, the interaction outputs the proposed commitment. In this case, both parties have been convinced that the hidden random variable is distributed according to $\mathcal{D}$, but only the Dealer knows the value of the random variable, which remains hidden to the Player for now. In this work, we will colloquially refer to this property as "random sample hiding." Else, the interaction outputs $\perp$.[21]

We note that from the Boolean circuit implementation of the trimmed-tree Knuth Yao algorithm, we can immediately construct random variable commitment schemes for arbitrary discrete distributions.

Theorem 6.1. *If $\mathcal{D}$ is a discrete distribution on bounded support, and there exists a random bit commitment scheme, then there exists a random variable commitment scheme for $\mathcal{D}_{b_p}$.*

Proof. Our construction takes as input the random bit commitments of [BGKW24] and applies the "witness-indistinguishable proof of knowledge" from Proposition 2.1 to execute the b_p-trimmed-tree Knuth-Yao boolean circuit on these input random bit commitments. By definition of witness indistinguishable proof of knowledge, this results in Knuth-Yao's output leaf label hidden within a commitment. Thereby we obtain random sample hiding and overall a sub-Gaussian commitment scheme, satisfying the conditions of Definition 7. □

A consequence of the minimal overhead in the PoK we use (Proposition 2.1) and the upper-bound on trimmed-tree Knuth-Yao boolean circuit complexity (Corollary 4.1), is that the RVCS is efficient for sub-Gaussian random variables.

Corollary 6.1. *Our RV commitment scheme construction applied to σ^2 sub-Gaussian random variables, for $\sigma\sqrt{b_p} \geq e^{-1/e}$ and $b_p \geq 2$, uses b_p random bit commitments and performs $O(c_{KY}) = O\left(\sigma \cdot b_t \cdot b_p^{3/2}\right)$ work on these, only reading each bit commitment once in the algorithmic execution path, to output a random-sample-hiding draw from $\mathcal{D}_{b_p}$ truncated on $[-t, t]$.*

Remark 4. *Theorem 4.2 implies that our RV commitment scheme construction will also be efficient for other discrete distributions with an appropriately bounded b_pth moment. Section 4.3 of [BGKW24] shows how to use an RV commitment scheme for $\mathcal{D}_{b_p}$ for an appropriate choice of b_p as a tool to construct certified probabilistic mechanisms which are additive in $\mathcal{D}$. In particular, Section 4.3 of [BGKW24] proves that this results in certified DP if done for a distribution $\mathcal{D}$ which gives a differentially private additive noise mechanism; this is the case for the b_t-truncated discrete Gaussian mechanism (Lemma 2.2). Next in Section 7 we prove that the parameters for $\mathcal{D}$ needed to achieve DP for counting queries over large datasets lead to a particularly efficient implementation with trimmed-tree Knuth-Yao.*

7 Knuth-Yao is Better For Private Mean Estimation Over Large Datasets

We now describe how making mean statistics private with the truncated discrete Gaussian mechanism, either over large domains or equivalently with high precision, favours the use of Knuth-Yao over rejection sampling. The ideas in this section can be abstracted to making arbitrary mechanisms with constant sensitivity private, where recall sensitivity is the maximum difference in outputs between neighbouring datasets. The primary idea for mean estimation is that if the datasets are sufficiently large, the sensitivity is ≤ 1 and so the σ of the sampled noise is small (independent of the size of the domain). Thus rejection sampling runs with $O(t)$

[21] If the interaction outputs $\perp$, repeated play to form a new commitment may be worthwhile, but care must be taken. The guarantees of RVCS allow the Dealer to safely engage in repeated play if the Player misbehaves, but the Player should be particularly cautious if they detect misbehavior from the Dealer. In general, we assume the Dealer has access to more computational power than the Player, so we can have particularly stringent expectations of their behavior. See Remark 4.1 of [BGKW24] for further discussion of this subtle asymmetry in RVCS.

cost for sampling the noise while Knuth-Yao runs with $O(\log(t))$ cost. We later remark on how finite-bit precision introduces error to the DP guarantees, which is the same regardless of the sampler.

Consider a dataset of size n of statistics varying from $[-t, t] \in \mathbb{Z}$. Note this could also be a discrete representation for statistics over a smaller interval in $\mathbb{R}$ but with some finite bits of precision; their number of bits of precision effectively grows the number of elements in the domain. Previous approaches to certified DP use the binomial mechanism [BGKW24], but the number of random bit commitments scales with $O(2^{b_t})$, only to provide DP guarantees analogous to the truncated discrete Gaussian. We instead will consider sampling from the truncated discrete Gaussian with the trimmed-tree Knuth-Yao algorithm to get exponential improvements. Recall that selecting the σ for the truncated discrete Gaussian to have DP requires knowing the "sensitivity" of the mean operation: $\Delta = \max_{\text{neigbouring}D,D'} \left|\sum_{x\in D}(x/n) - \sum_{x\in D'}(x/n)\right|$.

Note that $\Delta \leq 2t/n$ as $|x - x'| \leq 2t \; \forall x, x' \in [-t, t]$. Now consider the σ needed to achieve $(1, 1/n) - DP$ (one usually takes $\delta \leq 1/n$ to avoid complete privacy leakage), which by Lemma 2.2 is $\sigma = \frac{2t}{n}\sqrt{2\ln(2n)}$. Given this, the number of iterations needed for rejection sampling is then $(2t+1)p(0) \geq \frac{2t+1}{\sqrt{2}\sigma+1} \geq \frac{2t+1}{2\frac{2t}{n}\sqrt{\ln(2n)}+1}$ where the first inequality follows from Fact 19 of [CKS20] noting the normalizing constant for the truncated discrete Gaussian is smaller than the discrete Gaussian. This gives the following corollary.

Corollary 7.1. *Picking σ to achieve $(1, \frac{1}{n})$-DP with the truncated discrete Gaussian for private mean estimation over $[-t, t]$ with a dataset of size n, the circuit complexity of rejection sampling to sample the noise grows with the dataset size:*

$$c_R \geq (b_p + b_t + c_p)\frac{2t+1}{2\frac{2t}{n}\sqrt{\ln(2n)}+1}$$

In contrast, Corollary 4.1 implies the trimmed-tree Knuth-Yao circuit complexity is $\leq O(\sigma b_p^{3/2} b_t) = O(\frac{2t}{n}\sqrt{2\ln(2n)}b_p^{3/2}b_t)$: it *decreases with the dataset size.*

Corollary 7.2. *Picking σ to achieve $(1, \frac{1}{n})$-DP with the truncated discrete Gaussian for private mean estimation over $[-t, t]$ with a dataset of size n, the circuit complexity of the trimmed-tree Knuth Yao algorithm to sample the noise decreases with the dataset size:*

$$c_{KY} \leq O\left(\frac{2t}{n}\sqrt{2\ln(2n)}b_p^{3/2}b_t\right)$$

Notably, after $\frac{2t}{n} \leq 1$ we may set the sensitivity to $= 1$ (suppose we only round to elements in $\mathbb{Z}_m$ after computing the mean with higher precision). Then rejection sampling has cost growing at least like $\frac{(2t+1)}{2\sqrt{\ln(2n)}+1}(b_p + b_t + c_p)$ but trimmed tree Knuth-Yao costs grows at most like $O(\sqrt{2\ln(2n)}b_p b_t)$. That is, *rejection sampling grows exponentially with the bits for the domain*, while *Knuth-Yao is linear*. Meanwhile, there is a modest $\sqrt{\ln(n)}$ cost Knuth Yao incurs with dataset size due to the δ factor for DP.

Finite-bit Error and DP guarantees We can now specify b_p by accounting for the error it introduces to the δ term, which recall from Corollary 3.1 is less than $9e^{1/e}\sigma\sqrt{m}2^{-\frac{b_p m}{m+1}}$ for any $m \geq 2$ sufficiently large that $(e^{1/e}\sigma m^{1/2})^{\frac{m}{m+1}} \geq 2^{-\frac{b_p}{m+1}}$. If we set σ for some δ' that is lower than some target δ, we can then match δ by setting b_p. For example, take $\delta' = \frac{1}{2}\delta$, and so given σ which gives $(\epsilon, \delta/2)$-DP, we have $b_p \geq \frac{1}{\ln(2)\frac{m}{m+1}}\ln\left(\frac{18e^{1/e}\sigma\sqrt{m}}{\delta}\right)$ suffices to have (ϵ, δ)-DP.

Remark 5. *The final comparison that trimmed tree Knuth-Yao has costs that grow at most like $c_{KY} \leq O(\sqrt{2\ln(2n)}b_p b_t)$ while rejection sampling has costs growing at least like $c_R \geq \Omega(\frac{(2t+1)}{2\sqrt{\ln(2n)}+1}(b_p + b_t + c_p))$ is in fact true for any function f whose DP-sensitivity is constant, i.e. $\Delta \leq O(1)$, when using an additive noise mechanism with the b_t-truncated discrete Gaussian to b_p-probability precision. For example, this also includes counting queries (i.e., the number of data subjects who have a property as defined by a predicate on their respective datapoint returning a binary value in $\{0, 1\}$) where the maximum change introduced by replacing a datapoint is always* 1.[22]

[22]Note that previously we were considering counting queries in the fractional averaging sense, as in [Vad17], but now we

References

[AA22] Amol Aggarwal and Josh Alman. Optimal-degree polynomial approximations for exponentials and gaussian kernel density estimation. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPIcs*, pages 22:1–22:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[BBE+19a] Gilles Barthe, Sonia Belaïd, Thomas Espitau, Pierre-Alain Fouque, Mélissa Rossi, and Mehdi Tibouchi. Galactics: Gaussian sampling for lattice-based constant- time implementation of cryptographic signatures, revisited. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 2147–2164, New York, NY, USA, 2019. Association for Computing Machinery.

[BBE+19b] Gilles Barthe, Sonia Belaïd, Thomas Espitau, Pierre-Alain Fouque, Mélissa Rossi, and Mehdi Tibouchi. Galactics: Gaussian sampling for lattice-based constant-time implementation of cryptographic signatures, revisited. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 2147–2164, 2019.

[BC23] Ari Biswas and Graham Cormode. Interactive proofs for differentially private counting. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 1919–1933, 2023.

[BGKW24] Zoë Ruha Bell, Shafi Goldwasser, Michael P Kim, and Jean-Luc Watson. Certifying private probabilistic mechanisms. In *Annual International Cryptology Conference*, pages 348–386. Springer, 2024.

[BLP22] Jay Bosamiya, Wen Shih Lim, and Bryan Parno. {Provably-Safe} multilingual software sandboxing using {WebAssembly}. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1975–1992, 2022.

[BLR+18] Shi Bai, Tancrède Lepoint, Adeline Roux-Langlois, Amin Sakzad, Damien Stehlé, and Ron Steinfeld. Improved security proofs in lattice-based cryptography: Using the rényi divergence rather than the statistical distance. *J. Cryptol.*, 31(2):610–640, 2018.

[BV17] Victor Balcer and Salil Vadhan. Differential privacy on finite computers. *arXiv preprint arXiv:1709.05396*, 2017.

[CDS94] Ronald Cramer, Ivan Damgard, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo G. Desmedt, editor, *Advances in Cryptology — CRYPTO '94*, pages 174–187, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.

[CFM08] Dario Catalano, Dario Fiore, and Mariagrazia Messina. Zero-knowledge sets with short proofs. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 433–450. Springer, 2008.

[CHL+05] Melissa Chase, Alexander Healy, Anna Lysyanskaya, Tal Malkin, and Leonid Reyzin. Mercurial commitments with applications to zero-knowledge sets. In *Advances in Cryptology–EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005. Proceedings 24*, pages 422–439. Springer, 2005.

[CKS20] Clément L Canonne, Gautam Kamath, and Thomas Steinke. The discrete gaussian for differential privacy. *Advances in Neural Information Processing Systems*, 33:15676–15688, 2020.

[CM99] Jan Camenisch and Markus Michels. Proving in zero-knowledge that a number is the product of two safe primes. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, pages 107–122, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

consider the bounded integer (e.g., binary $\{0, 1\}$) summing sense, as in [BGKW24].

[CSSV23] Aloni Cohen, Adam Smith, Marika Swanberg, and Prashant Nalini Vasudevan. Control, confidentiality, and the right to be forgotten. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 3358–3372, 2023.

[DCCM22] Jiaxin Deng, Simin Chen, Jiageng Chen, and Weizhi Meng. A survey on discrete gaussian samplers in lattice based cryptography. In *International Symposium on Emerging Information Security and Applications*, pages 87–107. Springer, 2022.

[DDLL13a] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In *Annual Cryptology Conference*, pages 40–56. Springer, 2013.

[DDLL13b] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013*, pages 40–56, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[Dev86] Luc Devroye. *Non-Uniform Random Variate Generation*, chapter 15: The Random Bit Model. Springer, 1986.

[DFW22] Yusong Du, Baoying Fan, and Baodian Wei. A constant-time sampling algorithm for binary gaussian distribution over the integers. *Information Processing Letters*, 176:106246, 2022.

[FS90] U. Feige and A. Shamir. Witness indistinguishable and witness hiding protocols. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, STOC '90, page 416–426, New York, NY, USA, 1990. Association for Computing Machinery.

[HHK+21] Yuncong Hu, Kian Hooshmand, Harika Kalidhindi, Seung Jin Yang, and Raluca Ada Popa. Merkle 2: A low-latency transparency log system. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 285–303. IEEE, 2021.

[HHM+05] Jason D Hartline, Edwin S Hong, Alexander E Mohr, William R Pentney, and Emily C Rocke. Characterizing history independent data structures. *Algorithmica*, 42:57–74, 2005.

[HPRR20] James Howe, Thomas Prest, Thomas Ricosset, and Mélissa Rossi. Isochronous gaussian sampling: from inception to implementation: With applications to the falcon signature scheme. In *Post-Quantum Cryptography: 11th International Conference, PQCrypto 2020, Paris, France, April 15–17, 2020, Proceedings 11*, pages 53–71. Springer, 2020.

[HRS+17] Andreas Haas, Andreas Rossberg, Derek L Schuff, Ben L Titzer, Michael Holman, Dan Gohman, Luke Wagner, Alon Zakai, and JF Bastien. Bringing the web up to speed with webassembly. In *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 185–200, 2017.

[JYCC+21] Hengrui Jia, Mohammad Yaghini, Christopher A Choquette-Choo, Natalie Dullerud, Anvith Thudi, Varun Chandrasekaran, and Nicolas Papernot. Proof-of-learning: Definitions and practice. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1039–1056. IEEE, 2021.

[KHF+20] Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, et al. Spectre attacks: Exploiting speculative execution. *Communications of the ACM*, 63(7):93–101, 2020.

[KRR+18a] Angshuman Karmakar, Sujoy Sinha Roy, Oscar Reparaz, Frederik Vercauteren, and Ingrid Verbauwhede. Constant-time discrete gaussian sampling. *IEEE Transactions on Computers*, 67(11):1561–1571, 2018.

[KRR+18b] Angshuman Karmakar, Sujoy Sinha Roy, Oscar Reparaz, Frederik Vercauteren, and Ingrid Verbauwhede. Constant-time discrete gaussian sampling. *IEEE Trans. Computers*, 67(11):1561–1571, 2018.

[KRVV19a] Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede. Pushing the speed limit of constant-time discrete gaussian sampling. a case study on the falcon sig-

nature scheme. In *Proceedings of the 56th Annual Design Automation Conference 2019*, pages 1–6, 2019.

[KRVV19b] Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede. Pushing the speed limit of constant-time discrete gaussian sampling. a case study on the falcon signature scheme. In *Proceedings of the 56th Annual Design Automation Conference 2019*, DAC '19, New York, NY, USA, 2019. Association for Computing Machinery.

[LSG+20] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, et al. Meltdown: Reading kernel memory from user space. *Communications of the ACM*, 63(6):46–56, 2020.

[LY10] Benoît Libert and Moti Yung. Concise mercurial vector commitments and independent zero-knowledge sets with short proofs. In *Theory of Cryptography Conference*, pages 499–517. Springer, 2010.

[Mau09] Ueli Maurer. Unifying zero-knowledge proofs of knowledge. In Bart Preneel, editor, *Progress in Cryptology – AFRICACRYPT 2009*, pages 272–286, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[MBB+15] Marcela S Melara, Aaron Blankstein, Joseph Bonneau, Edward W Felten, and Michael J Freedman. {CONIKS}: Bringing key transparency to end users. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 383–398, 2015.

[Mer87] Ralph C Merkle. A digital signature based on a conventional encryption function. In *Conference on the theory and application of cryptographic techniques*, pages 369–378. Springer, 1987.

[Mic97] Daniele Micciancio. Oblivious data structures: applications to cryptography. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 456–464, 1997.

[Mir12] Ilya Mironov. On significance of the least significant bits for differential privacy. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 650–661, 2012.

[MRK03] Silvio Micali, Michael Rabin, and Joe Kilian. Zero-knowledge sets. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 80–91. IEEE, 2003.

[NT01] Moni Naor and Vanessa Teague. Anti-persistence: History independent data structures. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 492–501, 2001.

[Pap94] Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

[Ped92] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, pages 129–140, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.

[SAB24] Megha Srivastava, Simran Arora, and Dan Boneh. Optimistic verifiable training by controlling hardware nondeterminism. *arXiv preprint arXiv:2403.09603*, 2024.

[SDS+18] Emil Stefanov, Marten van Dijk, Elaine Shi, T-H Hubert Chan, Christopher Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas. Path oram: an extremely simple oblivious ram protocol. *Journal of the ACM (JACM)*, 65(4):1–26, 2018.

[Spi71] Philip M. Spira. On the time necessary to compute switching functions. *IEEE Trans. Computers*, 20(1):104–105, 1971.

[STC+24] Ali Shahin Shamsabadi, Gefei Tan, Tudor Ioan Cebere, Aurélien Bellet, Hamed Haddadi, Nicolas Papernot, Xiao Wang, and Adrian Weller. Confidential-dpproof: Confidential proof of differentially private training. In *International Conference on Learning Representations (ICLR)*, 2024.

[SU] Adam Smith and Jonathan Ullman. Privacy in statistics and machine learning spring 2021 lecture 9: Approximate differential privacy. `https://dpcourse.github.io/2021-spring/lecnotes-web/lec-09-gaussian.pdf`.

[SZJ+22] Shuo Sun, Yongbin Zhou, Yunfeng Ji, Rui Zhang, and Yang Tao. Generic, efficient and isochronous gaussian sampling over the integers. *Cybersecurity*, 5(1):10, 2022.

[Tha22] Justin Thaler. Proofs, arguments, and zero-knowledge. *Found. Trends Priv. Secur.*, 4(2-4):117–660, 2022.

[TJSP22] Anvith Thudi, Hengrui Jia, Ilia Shumailov, and Nicolas Papernot. On the necessity of auditable algorithmic definitions for machine unlearning. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 4007–4022, 2022.

[Vad17] Salil P. Vadhan. The complexity of differential privacy. In Yehuda Lindell, editor, *Tutorials on the Foundations of Cryptography*, pages 347–450. Springer International Publishing, 2017.

[vN51] John von Neumann. Various techniques used in connection with random digits. In *J. Research Nat. Bur. Stand., Appl. Math. Series*, pages 12:36–38, 1951.

[WPH+22] Yingchen Wang, Riccardo Paccagnella, Elizabeth Tang He, Hovav Shacham, Christopher W Fletcher, and David Kohlbrenner. Hertzbleed: Turning power {Side-Channel} attacks into remote timing attacks on x86. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 679–697, 2022.

[WYF+23] Chengkun Wei, Ruijing Yu, Yuan Fan, Wenzhi Chen, and Tianhao Wang. Securely sampling discrete gaussian noise for multi-party differential privacy. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 2262–2276, 2023.

A Appendix: From Variance Factor to Lipschitz from Section 4.3

Corollary A.1. *For the b_t-truncated discrete Gaussian with $\sigma \geq (2^{b_p+1}\sqrt{b_p+2}\frac{\sqrt{\ln(2)}e}{\sqrt{\pi}})^{\frac{1}{2}}+1$, with $t \geq \sigma\sqrt{\ln\sigma}$, the size of its b_p-bit precision DDG tree is*

$$s_{b_p} \geq \Omega\left(\sigma + b_p - \log(\sigma)\right).$$

Proof. This will follow from Corollary 4.3, by showing $L \leq 2^{-b_p}$ under the assumptions. Note by Lemma 2.3 with the DP guarantees stated in Lemma 2.2 we have $L \leq (e^{\frac{\sqrt{2\ln(2/\delta)}}{\sigma}}-1)p(0)+\delta$ for some δ. Take $\delta \leq 2^{-b_p-1}$, it then suffices to show $(e^{\frac{\sqrt{2\ln(2)(b_p+2)}}{\sigma}}-1)p(0) \leq 2^{-b_p-1}$.

Recall from the proof of Corollary 4.3 that $p(0) \leq \frac{1}{\sqrt{2\pi}(\sigma-1)}$ for our condition on t. Further note by the monotonicity of the exponential derivative that $(e^{\frac{\sqrt{2\ln(2)(b_p+2)}}{\sigma}}-1) \leq e^{\frac{\sqrt{2\ln(2)(b_p+2)}}{\sigma}}\frac{\sqrt{2\ln(2)(b_p+2)}}{\sigma}$. Thus it further suffices to show $e^{\frac{\sqrt{2\ln(2)(b_p+2)}}{\sigma}}\frac{\sqrt{2\ln(2)(b_p+2)}}{\sigma}\frac{1}{\sqrt{2\pi}(\sigma-1)} \leq 2^{-b_p-1}$. Note $e^{\frac{\sqrt{2\ln(2)(b_p+2)}}{\sigma}} \leq e$ for our choice of σ, and that $\frac{\sqrt{2\ln(2)(b_p+2)}}{\sigma}\frac{1}{\sqrt{2\pi}(\sigma-1)} \leq 2^{-b_p-1}\frac{1}{e}$. So we conclude for $\sigma \geq (2^{b_p+1}\sqrt{b_p+2}\frac{\sqrt{\ln(2)}e}{\sqrt{\pi}})^{\frac{1}{2}}+1$ that $L \leq e^{\frac{\sqrt{2\ln(2)(b_p+2)}}{\sigma}}\frac{\sqrt{2\ln(2)(b_p+2)}}{\sqrt{2\pi}(\sigma-1)\sigma}+2^{-b_p-1} \leq 2^{-b_p}$.

□

B Appendix: Proof of Corollary 4.3

Proof. This follows from the previous theorem if we show $\frac{1}{p(0)} = \sum_{x=-t}^{t} e^{-x^2/2\sigma^2} \geq \Omega(\sigma)$. Note from Fact 19 of [CKS20] that $\sum_{\mathbb{Z}} e^{-x^2/2\sigma^2} \geq \sqrt{2\pi}\sigma$ and $\sum_{x\leq -t} e^{-x^2/2\sigma^2} + \sum_{x\geq} e^{-x^2/2\sigma^2} \leq \mathbb{P}(|x| \geq t)(\sqrt{2\pi}\sigma+1) \leq \frac{\sqrt{2\pi}\sigma+1}{\sigma} \leq \sqrt{2\pi}$ where the first inequality also follows from Fact 19 of [CKS20] to bound the normalizing

constant over $\mathbb{Z}$, and the second inequality follows from Fact 1 using the condition on t, and as $\sigma \geq 1$. Together this implies $\sum_{x=-t}^{t} e^{-x^2/2\sigma^2} \geq \sqrt{2\pi}\sigma - \sqrt{2\pi} \geq \Omega(\sigma)$.

□