MARCIO BARBADO JUNIOR

Efficient Gaussian sampling for the construction of lattice-based post-quantum cryptosystems

Revised Version

Dissertation submitted to Escola Politécnica da Universidade de São Paulo fulfilling the requirements for the degree of Master of Science.

São Paulo
2023

MARCIO BARBADO JUNIOR

Efficient Gaussian sampling for the construction of lattice-based post-quantum cryptosystems

Dissertation submitted to Escola Politécnica da Universidade de São Paulo fulfilling the requirements for the degree of Master of Science.

Concentration area: Computer Engineering

Supervisor: Pedro Luiz Pizzigatti Corrêa

São Paulo
2023

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 18 de Dezembro de 2023.

Assinatura do autor:

Assinatura do orientador:

Catalogação-na-publicação

Barbado Junior, Marcio

Amostragem gaussiana eficiente para a construção de sistemas criptográficos pós-quânticos baseados em reticulados/ M. Barbado Junior -- versão corr. -- São Paulo, 2023.

120 p.

Dissertação (Mestrado) – Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais (PCS).

1. Criptologia. 2. Aprendizado com erros em anel. 3. Amostragem gaussiana discreta. 4. Teorema central do limite. 5. Transformada rápida de Walsh–Hadamard. I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais (PCS). II. t.

ACKNOWLEDGMENTS

Not without efforts I got this masters. As one can infere, starting to pursue such degree with 42 has required me to build a self-motivation foundation to begin with. An even harder task amid the global COVID-19 pandemic, and the very own life at fourty two. At some point during my lattice-based cryptography learning process, weird associated subjects started to pop up, and that has put me in a confusion state. I remember reading the expression “ring learning with errors”, and thinking, as of where that had came from. It took me some time to figure out semantics, only to realize the bigger challenge before me. Luckily, the studied subject was fascinating enough, and also, I was surrounded by relevant people and organizations with valuable things to share, be it of emotional, technical or financial nature.

For emotional support, thank you, beloved daughter Beatriz and beloved son Paulo. You have helped me in many ways. I’ve always had my motivation reaffirmed when we were together. I am also grateful for having been raised by my parents, who provided me with strength to face troublesome situations like this challenging adventure. Also, attending therapy sessions has proven to be effective in making self-motivation flourish. Thus, thank you, Mateus Pinheiro, for optimizing my insanity.

Additionally, people endowed with outstanding knowledge has supported me in this work. Thank you, Prof. Dr. Pedro Luiz Pizzigatti Corrêa, Prof. Dr. Marcos Simplicio, Prof. Dr. Átila Madureira Bueno, and Prof. Dr. Anarosa Alves Franco Brandão, for offering me the amazing opportunity to walk and complete this memorable path. LARC colleagues Marcos Vinicius, Erina and Renan shared pleasant moments and invaluable technical remarks with me. Plus, for helping me with all of the paperwork in the final steps of this journey, and also with many reminders, Mariza Ushijima Leone, from the PCS office, just became my new favorite hero. Thank you, guys. I am also grateful for the insightful observations of Prof. Dr. Routo Terada, which stimulated me to think and consider different aspects of the problems in this work. Dr. Geovandro Pereira provided this work with a tremendous upgrade in technical quality, due to his careful and thorough review. As a reputed professional, Dr. Jefferson Evandi Ricardini Fernandes de Oliveira enriched this work with his wide experience in lattice-based cryptography, remarking valuable security information points. Even the essential aspects of lattice theory are covered herein, due to the great contributions of Prof. Dr. Kacper Pluta, who shared his notable knowledge on geometry. Cieszę się, że mój drogi przyjaciel Kacper Pluta kazał mi zaakceptować LaTeX.

Likewise, equally noteworthy technical improvements were possible thanks to the anonymous reviewers of SBSeg and WPGEC.

Last but not least, for financial support, I would like to thank CNPq, FDTE and FUSP.

Gratitude to all of you.

RESUMO

Com o avanço da computação quântica, a segurança computacional de esquemas criptográficos assimétricos clássicos amplamente utilizados, como os baseados em problemas de fatoração de inteiros e logaritmo discreto, encontra-se sob ameaça. Por essa razão, pesquisadores na área de criptografia têm buscado esquemas alternativos resistentes a ataques quânticos. Nesse cenário, abordagens criptográficas tais quais as baseadas em teoria de reticulados, que até então eram menos usadas por serem consideradas computacionalmente custosas, recuperam a atenção dos criptógrafos, e passam a figurar como opções viáveis. Este trabalho tem como objetivo contribuir para consolidar essa retomada na adoção da abordagem baseada em reticulados. Especificamente, tem-se como alvo a formulação do problema de aprendizado com erros em anel (*ring learning with errors*), aqui usada para a produção de erro, o que propicia a criação de chaves criptográficas supostamente mais seguras. As referidas chaves são formadas como polinômios de coeficientes produzidos através de amostragem, realizada em uma função de probabilidade associada a uma distribuição gaussiana. A construção de amostradores dessa natureza é parte da maioria dos projetos criptográficos baseados em reticulados, e frequentemente representa duas barreiras principais: um gargalo de eficiência, e um risco de vazamento de informação devido a ataques de canal colateral baseados em temporização. Procura-se reduzir a barreira de ineficiência através de técnicas para a aceleração da convergência do teorema central do limite durante as criações de distribuições normais, e também através do emprego da transformada rápida de Walsh–Hadamard para a geração de valores aleatórios. Já o vazamento de informação por ataques de temporização tem seu risco atenuado pela implementação (em software) da primitiva como um gerador de números aleatórios com rotinas isócronas. Métricas estatísticas clássicas empregadas mostram os benefícios do esquema e sua adequação, quando comparado a uma amostragem gaussiana discreta baseada na tabela de distribuição acumulada, aqui considerado o método de referência, dada a sua adoção em diversos esquemas criptográficos baseados em reticulados. Testes com até 2^{23} amostras são conduzidos, e os resultados são favoráveis ao amostrador aqui apresentado.

Palavras-chave: Criptologia. Aprendizado com erros em anel. Amostragem gaussiana discreta. Teorema central do limite. Transformada rápida de Walsh–Hadamard.

ABSTRACT

As quantum computing advances, the computational security of widely adopted classic cryptographic schemes, like the asymmetric ones based on integer factorization and discrete logarithm problems, is put at risk. For that reason, cryptography researchers seek alternatives to resist quantum attacks. In that scenario, cryptographic approaches like the one based on lattice theory, mostly despised until then for being considered too computationally costly, regain the attention of cryptographers as viable alternatives. This work aims at contributing to consolidating that lattice-based approach resumption. Specifically, focus is given to the ring learning with errors problem formulation, explored herein for artificial noise generation, which propitiates stronger cryptographic keys. The referred keys are built as polynomials, whose coefficients are produced through sampling from a probability mass function, associated with a truncated normal distribution. Crafting a sampler like that, called Gaussian, is a part of most lattice-based cryptographic projects, and it often imposes two major barriers: an efficiency bottleneck, and an information leakage risk due to side-channel timing attacks. Part of the efficiency problem is overcome by accelerating convergence of the central limit theorem in the creation of a normal distribution, and by obtaining samples through a fast Walsh–Hadamard transform strategy. As for the information leakage risk, it is mitigated via software implementation of isochronous routines in the random number generator. Classic statistical metrics are employed to show the advantages and suitability of the scheme, when compared to a cumulative distribution table sampler, here considered as a reference, given the fact it is used in many lattice-based cryptographic schemes. Tests with up to 2^{23} sampling queries are conducted, and the results are favorable to the sampler presented herein.

Keywords: Cryptology. Ring learning with errors. Discrete Gaussian sampling. Central limit theorem. Fast Walsh–Hadamard transform.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS AND ACRONYMS

AES	Advanced Encryption Standard
BDD	Bounded Distance Decoding
BKZ	blockwise Korkine-Zolotarev
BLISS	Bimodal Lattice Signature Scheme
CDT	cumulative distribution table
CLI	command line interface
CLT	Central Limit Theorem
CPU	central processing unit
CRYSTALS	cryptographic suite for algebraic lattices
CVP	closest vector problem
DDG	discrete distribution generating
DFT	discrete Fourier transform
DGS	discrete Gaussian sampling
DH	Diffie–Hellman
DiGS	discrete Gaussian sampler
DSA	Digital Signature Algorithm
ECC	elliptic curve cryptography
ECDSA	elliptic curve digital signature algorithm
EPUSP	Escola Politécnica da Universidade de São Paulo

FALCON	fast Fourier lattice-based compact signatures over NTRU
FFT	fast Fourier transform
FLOPS	floating point operations per second
FWHT	fast Walsh–Hadamard transform
GCC	GNU Compiler Collection
GGH	Goldreich, Goldwasser, and Halevi
GMP	GNU Multiple Precision Arithmetic Library
GSL	GNU Scientific Library
IoT	Internet of Things
IV	initialization vector
LLL	Lenstra–Lenstra–Lovász
LWE	learning with errors
MLE	maximum likelihood estimator
Module-LWE	module learning with errors
Module-SIS	module short integer solution
NIST	National Institute of Standards and Technology
NP	nondeterministic polynomial
P2P	peer-to-peer
PDF	probability density function
PKI	public-key infrastructure
PMF	probability mass function
PoC	proof of concept

PPGEE	graduate program in electrical engineering
PRNG	pseudo-random number generator
r.v.	random variable
Ring-LWE	ring learning with errors
RLWE	ring learning with errors
RNG	random number generator
RSA	Rivest Shamir Adleman
SHA	Secure Hash Algorithm
SIS	short integer solution problem
SIVP	shortest independent vector problem
SVP	shortest vector problem
uSVP	unique shortest vector problem
USP	Universidade de São Paulo

LIST OF SYMBOLS

β an exponent responsible for determining the number of coefficients each polynomial is formed of, e.g., a polynomial formed by 2^{β} coefficients.

η lowercase eta is a matrix, specifically the initialization vector to be transformed.

μ lowercase mu is the mean for a given normal probability distribution.

σ lowercase sigma is the standard deviation for a given normal distribution.

σ^2 the variance for a given normal distribution.

τ lowercase tau is the tail-cut factor for a given normal distribution.

D the random variable of a normal distribution, discrete case.

$\mathcal{D}$ normal distribution, discrete case.

$g(\omega)$ a mathematical function of frequency.

ℓ the number of cycles on a given stage of the fast Walsh–Hadamard transform algorithm presented herein.

N the random variable of a normal distribution, continuous case.

$\mathcal{N}$ normal distribution, continuous case.

O the origin of an Euclidean space.

$P(x)$ a probability function.

R_q finite commutative ring of polynomials modulus q.

s the sample size.

S Gaussian parameter.

CONTENTS

1 INTRODUCTION

This work addresses efficiency and security issues related to post-quantum cryptography or quantum-safe cryptography, i.e., cryptographic schemes that are supposedly resistant against quantum computer attacks. Such schemes can be built upon different types of computational problems. This work is interested in problems arising from lattice theory (GRÄTZER, 2011; WEHRUNG et al., 2016), which form what is called the lattice-based cryptography.

As for this introductory chapter, it firstly presents historical moments regarding the subject, which support the following section about motivation. The motivation section also includes a short description of the problem involved, which completes the stimulus explanation. Next section presents the goal, followed by related works and method. Finally, as our research report enforces the style of presenting something presumably better, this chapter concludes in Section 1.6 with a brief description of the contribution herein provided.

1.1 A brief history of lattice-based cryptography

Until circa 1990, mathematical theories supporting lattices were mostly used in cryptology for cryptanalytic purposes (ODLYZKO, 1990). That scenario changed after Peter Shor described polynomial-time quantum solvers able to deal with widely-adopted problems (SHOR, 1994; SHOR, 1996), namely the discrete log problem and the factoring problem. The discrete log problem serves as the basis for elliptic-curve cryptography (ECC) (KOBLITZ, 1987) and for the Digital Signature Algorithm (DSA) standard (KERRY; GALLAGHER, 2013), whereas the factoring problem serves as the basis for RSA (RIVEST; SHAMIR; ADLEMAN, 1978). What the solvers of Shor represent since then are risks to the asymmetric cryptography practiced in regular computers. Such risks are inflicted by the supposed existence of quantum computers. Then, quantum-resistant cryptographic approaches proved to be necessary.

In 1996, an answer addressing the referred theoretical adversary is provided by (AJTAI, 1996), which presented a quantum-safe lattice-based exploration of a problem known as short integer solution (SIS). It served as the basis for the NTRU public-key cryptosystem (HOFFSTEIN; PIPHER; SILVERMAN, 1998).

In the year 2000, an NTRU patent application is granted in the United States of America (HOFFSTEIN; PIPHER; SILVERMAN, 2000; HOFFSTEIN; PIPHER; SILVERMAN, 2020). Later on, a few relevant studies arose, related to that new kind of cryptography, e.g., the NTRU-like cryptosystem known as GGH (GOLDREICH; GOLDWASSER; HALEVI, 1997). Though, the machines supposed to take advantage of the quantum routines described by Shor seemed distant from reality, and advancements occurred slowly. In 2008, more than a decade after (AJTAI, 1996), a standard from the Institute of Electrical and Electronics Engineers (IEEE) corroborated the lattice-based alternative (WHYTE et al., 2008). In the same year, a reference book covering quantum-resistant cryptographic approaches, including lattice-based ones, is published (BERNSTEIN et al., 2008). The referred group of approaches are generically called post-quantum.

As quantum computing approached feasible implementations, the threat of quantum attacks became imminent, and in 2016, the National Institute of Standards and Technology (NIST), a United States federal agency, publish a standardization call for post-quantum cryptography candidates (KIMBALL, 2016; CSRC, 2019). As a result, a fierce competition took place [1].

1.2 Motivation of this work

Given the relevance of lattice-based cryptography, and its learning with errors computation problem (LWE) (REGEV, 2009), it is appropriate to focus research efforts for the benefit of asymmetric cryptography. As mentioned, this work is about the LWE problem, precisely in its algebraically structured variant named ring learning with errors (RLWE) (LYUBASHEVSKY; PEIKERT; REGEV, 2013b; PEIKERT; PEPIN, 2019). The LWE problem consists in distinguishing between two types of equations: truly random ones, and those whose underlying structure have been masked by some controlled amount of noise. The RLWE problem has an analogous formulation, the main difference being that the underlying structure hidden involves polynomial rings. The interest in the RLWE problem lies in the fact that, besides being conjectured to be as hard as LWE for quantum computers, it also facilitates the construction of efficient cryptographic schemes. Not surprisingly, the RLWE assumption is adopted by many submissions to the NIST competition, as further discussed in Section 1.4. Still, it has some relevant shortcomings. Often, the generation of random values in RLWE-based cryptosystems make use of discrete Gaussian sampling (DGS) routines, which may suffer from efficiency issues (DWARAKANATH; GALBRAITH, 2014; ORTIZ, 2016), and may be vulnerable to timing attacks (BERNSTEIN et al., 2008; ORTIZ, 2016; ALKIM et al., 2019; PÖPPELMAN et al., 2019; ZHAO; STEINFELD; SAKZAD, 2020).

[1] A short description of the NIST post-quantum cryptography standardization process is available in Section A.1.

1.3 Goal

The goal is to elaborate a new and competitive discrete Gaussian sampling strategy for lattice-based cryptography. Firstly, this new DGS strategy intends to offer more efficient parameter setups, which intends to be accomplished in a relative manner, through comparison against a widely-accepted reference DGS strategy. Such comparison is established through measures of central processing unit (CPU) cycles. Secondly, the DGS construction presented in this work intends to be resistant against timing side-channel attacks. Evidences of the need for efficiency and security improvements in the DGS primitive are being provided by many state-of-the-art works, mostly stimulated by the post-quantum cryptography standardization process being promoted by NIST (CSRC, 2019). Both the justification for the DGS efficiency goal, and the justification for addressing the referred side-channel vulnerability, are provided in 1.2. Lastly, it is also in interest of this work to publish obtained results, so as to contribute with the cryptography community.

1.4 Related works

The following paragraphs present the main sources this work benefits from. It starts presenting essential lattice-based cryptography material, which paves the way for the specific subjects used here. Following, it focus on LWE and RLWE works, and then in discrete Gaussian sampling, which constitutes a major source of inefficiency problems for LWE and RLWE cryptographic schemes. Next, the works on the Central Limit Theorem (CLT) are presented, for that subject is explored as a means of achieving advantageous efficiency. Another subject whose related works are presented is the fast Walsh–Hadamard transform (FWHT), which connects all other topics, since it is used here as the structure of an alternative lattice-based DGS strategy, and it also stands as a resourceful artifice towards efficiency improvements.

1.4.1 Lattice theory in cryptography

A panorama of lattice-based cryptography previous research efforts in Brazil is given in (ORTIZ; ARANHA; DAHAB, 2015; BARGUIL, 2015; ORTIZ, 2016).

Recent efforts focusing on the trade-off between efficiency and security for state-of-the-art lattice-based schemes can be found in (ALBRECHT et al., 2018; ALBRECHT et al., 2019), which offer comparison benchmarks, and the possibility of simulating a few parameter combinations for NIST candidate submissions. The latter works helped in modeling the Gaussian sampling approach hereby presented, as further discussed in Section 3.2.

1.4.2 Learning with errors and ring learning with errors

In 2009, the learning with errors (LWE) problem formulation was presented to the world (REGEV, 2009). A few months after, the same author, Regev, along with Lyubashevsky and Peikert came up with the first formulation of the ring learning with errors (RLWE) problem, which has an updated text in (LYUBASHEVSKY; PEIKERT; REGEV, 2013b). In the following year,(BARRETO et al., 2014) was published, consisting in a chapter whose information is presented in a straightforward manner, assisting the present work. Lastly, a fortunate presentation helped in understanding how the noise vector is mathematically constructed (DING, 2019).

1.4.3 Discrete Gaussian sampling in lattice-based cryptography

Comprehensive web pages, listing works about discrete Gaussian sampling (DGS) in lattice-based cryptography can be found on (MICCIANCIO, 2019; BERNSTEIN et al., 2021). A historical perspective of optimizations is available in (FOLLÁTH, 2014). Prominent lattice-based cryptography works that rely on DGS are the Bimodal Lattice Signature Scheme (BLISS) digital signature scheme (DUCAS et al., 2013), the qTESLA digital signature scheme (ALKIM et al., 2019; AKLEYLEK et al., 2019), the FrodoKEM key encapsulation or key exchange scheme (ALKIM et al., 2020), and the Falcon digital signature scheme (HOWE et al., 2019; FOUQUE et al., 2020). BLISS uses the SIS problem assumption, and its DGS is based on inversion and rejection. The qTESLA scheme uses the RLWE problem assumption, and its DGS is based on CDT and rejection. Falcon uses a SIS over NTRU problem assumption, and its sampling strategy is based on the fast Fourier transform, i.e., it uses a trapdoor sampler, which requires floating-point arithmetic, but it also relies on DGS with CDT and rejection. Those related works comprise cryptographic schemes with embedded samplers.

There are also related works about standalone samplers. It is the case of the GALACTICS sampler (BARTHE et al., 2019), and the FACCT sampler (ZHAO; STEINFELD; SAKZAD, 2020), both being based on the BLISS embedded sampler. It is also the case of (SUN et al., 2021), which offers an alternative sampler for Falcon. Besides the standalone samplers inspired by previously existing cryptographic schemes, there is COSAC (ZHAO; STEINFELD; SAKZAD, 2019), a Gaussian sampler from the creators of FACCT, which is not based on previously existing works.

Additionally,(WANG; LYU; LIU, 2019) presents the DGS topic from an information theory perspective, and (CHEN, 2019) approaches the decision versions of RLWE with possible DGS primitives, questioning the hardness for some cyclotomic number fields and specific DGS interval widths. Further insights on DGS implementation, regarding generic and specific prerequisites, e.g., isochrony and CDT implications, are offered by (REPARAZ; BALASCH; VERBAUWHEDE, 2016; KARMAKAR et al., 2018; MICCIANCIO; WALTER, 2018; WANG; LING, 2019), and also by two C language implementations, namely the GNU Scientific Library (GSL) (GALASSI; THEILER, 2019), and the Open Quantum Safe library (liboqs) (STEBILA; MOSCA, 2021).

A comparison of sampling strategies among related works is present in Table 1.

The hereby proposed work is referred to as "DiGS" in the table, which is short for "discrete Gaussian sampling".

Table 1: DGS base strategies for DiGS and related works.

Sampler	**Assumption**	**DGS strategy**
DiGS (ours)	RLWE	FWHT
BLISS (DUCAS et al., 2013)	SIS	Inversion and rejection
FACCT (ZHAO; STEINFELD; SAKZAD, 2020)	generic	Inversion and rejection via CDT
Falcon (FOUQUE et al., 2020)	SIS over NTRU	CDT and rejection
FrodoKEM (ALKIM et al., 2020)	LWE	CDT
qTESLA (ALKIM et al., 2019)	RLWE	CDT and rejection

Source: author (2023).

1.4.4 General central limit theorem

Some efforts in the present work are motivated by (DWARAKANATH; GALBRAITH, 2014), which constitutes one of its initial influences, regarding the use of the general central limit theorem (CLT). Then, (BHARUCHA-REID; SAMBANDHAM, 1986) added valuable information about the relationships among the CLT, the number of zeros in random polynomials, and also the products and sums of random companion matrices. Resources for getting to understand the CLT formally are (BILLINGSLEY, 1961; BILLINGSLEY, 1995; COVER; THOMAS, 2006). Those works present a number of CLT enunciations, e.g., the Lindeberg–Lévy, and the Lyapounov theorems, both assuming probability distributions in asymptotic contexts, but independent random variables with high probability of being small. Insights are provided by (BERRY, 1940), which affords relevant roles to both the second and third order absolute moments of a given Gaussian distribution, respectively variance and skewness, regarding its least upper bound. Earlier works seeking applications of the CLT for fast generation of Gaussian random variables were found in (RADER, 1969). In it, the author presents an Hadamard transform sampling strategy for a context in which many random variables are necessary, but only one is obtained by conventional means.

1.4.5 The fast Walsh–Hadamard transform as a shuffling strategy

Regarding the fast Walsh–Hadamard transform (FWHT), didactic resources to support this work are (SYLVESTER, 1867; PRATT; KANE; ANDREWS, 1969; RADER, 1969; HARWIT; SLOANE, 1979; EVANGELARAS; KOUKOUVINOS; SEBERRY, 2003; LU et al., 2013). Hadamard matrices and the Sylvester method originate in (SYLVESTER, 1867). The FWHT is detailed in 2.4.1.

Random variable generation by means of the Hadamard matrix is found both in (RADER, 1969) and in the appendix of (HARWIT; SLOANE, 1979), the latter being more didactic. Besides, it also covers the relationships between Hadamard matrices and Walsh functions, the Hadamard transform, and its fast variant. In (EVANGELARAS; KOUKOUVINOS; SEBERRY, 2003), it is also possible to study how Hadamard matrices relate to Walsh functions. Additionally, the referred work suggests FWHT as a potential helper in speeding up Hadamard transforms.

In the search for efficiency, even the image compression subject was studied. As a matter of fact, FWHT is a long known solution for such purpose (PRATT; KANE; ANDREWS, 1969).

Finally, the work in (LU et al., 2013) has contributed to arouse interest in the FWHT. The referred article uses the transform in an alternative ridge regression algorithm, and results are compared against other constructions, e.g., a general ridge regression algorithm, a standard principal component algorithm, and a randomized principal component algorithm. Comparisons are established, based upon measures of floating point operations per second (FLOPS). Concerning the general ridge regression algorithm, computational gains of 70 percent are reported, favoring the FWHT approach. The proposed construction is also favored against the principal component algorithms in all of the tested setups. Computational gains of at least 82 percent are reported.

1.5 Method

For this work, a scientific method is enforced. It assumes there are efficient discrete Gaussian sampling parameter setups, which take advantage of a fast Fourier transform (FFT) in its sampling strategy. Namely, the FFT used herein is the fast Walsh–Hadamard transform (FWHT). In the pursuit of the goal presented in Section 1.3, computer routines are then built to test the referred hypothesis, according to Chapter 3. As reference for comparison purposes, a sampler based upon cumulative distribution table (CDT) is used, due to CDT being adopted by many state-of-the-art lattice-based schemes. Further details are present in Table 1.

1.5.1 Steps

The experimental approach is organized in steps, corresponding to the next enumerated item list, all of them being eligible to suffer interventions by the author.

i) Problem identification.

   Mostly built upon critical reading and reasoning, added to discussion activities with supervisor and other students.

ii) Hypothesis and prediction.

   As presented in Section 1.4, the fast Walsh–Hadamard transform (FWHT), and the Central Limit Theorem (CLT) can be used to improve the efficiency of DGS. In this step, the mathematics to support this work is elaborated, and an initial parameter setting, described in Section 3.2 is defined.

iii) Prototyping.

   Mathematical formalization, produced by hypothesis and prediction, is implemented in isochronous computer routines. Initial values for input variables, including the initial probability density function (PDF), follow the initial parameter setting present in Section 3.2. Then, two distinct sampling strategies are implemented, a CDT-based, which is the default trustworthy construction for comparisons, and an FWHT-based, which is the one presented by this work.

iv) First sampling.

Computer routines implemented in the prototyping step are used to generate data by means of sampling. This is done by sampling the initial PDF with both CDT and FWHT strategies, in order to build two probability mass functions. The probability mass function (PMF) is a discrete version of a probability density function (PDF). Next, the CDT PMF and the FWHT PMF are compared with the algebraic PDF. In the scope of this work, that is called a first-level comparison One bad PMF invalidates the process, which should then start over.

v) Second sampling.

The qualified CDT PMF and FWHT PMF are tested and compared against each other. In the scope of this work, that is called a second-level comparison The metrics of quality and efficiency are obtained from output variables, described in 3.1, and the results are organized for the subsequent, final step.

vi) Discussion.

This is an analytical step, performed on the generated data, regarding the goal of this work. Additional insights and remarks on this work as a whole may be produced. Also, in this step, an article is prepared and submitted to publication.

Lastly, it is noted that the first-level and second-level comparisons stand as falsifiability tests (POPPER, 2005).

1.6 Contribution

The discrete Gaussian sampling construction presented by this work is efficient and isochronous. It is shown that carefully chosen parameters will make a fast Walsh–Hadamard transform sampler more efficient and more precise than a typical CDT-based construction. Additionally, its isochronous routines make the sampler side-channel resistant.

1.7 Document structure

Beyond the present chapter, the following structure is applied to this document.

Chapter two covers the technical building blocks for the Gaussian sampler, i.e., a few mathematical concepts, the RLWE problem, lattice-based cryptography, and side-channel attacks. The mathematical concepts include algebra, calculus and probability, e.g., polynomial rings, lattice theory, the discrete Fourier transform, the Hadamard matrix, the fast Walsh–Hadamard transform, normal distribution, sampling algorithms, the central limit theorem, and the metrics adopted herein.

Chapter three presents specifications and implementation details regarding the Gaussian sampler, i.e., its variables, modeling, and computer routine specifications.

Tests and results are presented in Chapter four, and discussed in Chapter five.

Appendix A presents full tables with values of metrics and Gaussian sampling results for both the CDT and FWHT routines compared in this work.

2 GENERAL CONCEPTS

Cryptology is referred here as the body of knowledge comprising both cryptography and cryptanalysis. In general, the former, which is the subject of this work, is a group of protocols, schemes, specifications, standards, and techniques, meant to be useful for authentication, privacy and integrity requirements. Since ancient times, cryptology have provided societies with solutions to their problems, and history has provided the concept of secret with a determinant role in making societies thrive or fall (KAHN, 1996).

Given the threats regarding information security (TERADA, 2008), the concepts of cryptology have been implemented in a variety of applications, and they go far beyond protecting money transfers and electronic government transactions. Solutions to diverse fields have cryptology in their cores, and some of the new quandaries to be confronted by modern-day societies can be satisfactorily addressed by astute cryptologic perspectives.

In networking, cryptology is embedded in state-of-the-art protocols (ION et al., 2019), and wireless sensors (MARGI et al., 2009; TSCHOFENIG; BACCELLI, 2019). Data exchange in the deluging Internet of Things (IoT) constitutes a new focus of major security concerns, as its variety of constrained devices creates a growing attack surface (TSCHOFENIG; BACCELLI, 2019).

Even abrupt cultural changes can benefit from appropriated cryptologic solutions. Take the 2019–2021 coronavirus pandemic dilemma, in which infected people needed to be rapidly recognized. Privacy-related issues rapidly arose from such context, as societies worldwide tried to figure out to which extent could privacy jeopardize public health. If exposure of medical data records pose as a reason of concern by one side, it may also form the solution to the other, and as the latter needed to be addressed, cryptology provided the world with balance by dealing with the contact tracing pandemic problem (ANDERSON, 2020; REICHERT; BRACK; SCHEUERMANN, 2020). Either by modelling old and new threats, or by managing their risks, the cryptology body of knowledge proves to be a significant actor in paving the road to an interconnected mankind.

Profusion of new paradigms seems to be inescapable (CONTE et al., 2017). Decades ago, computers have led to the development of cryptography, and nowadays, the embryonic phase of quantum computing triggers a new wave of research initiatives whereby improvement and recycling for this area of knowledge is sought. Many researchers think post-quantum cryptography, also called quantum-resistant or quantum-safe cryptography, constitutes such recycling. It should be noted the post-quantum designation has been used for some time, and it was even adopted in the NIST standardization process, the reasons why the referred designation seems to occur more often (CSRC, 2019). This new area of study refers to cryptographic approaches based upon problems which are supposedly harder to be solved by quantum computers than those based on integer factoring, or discrete logarithm.

In the scope of this work, the main subjects of mathematics used are algebra, calculus, and probability. Those three subjects, along with cryptology itself form the structuring sections for this chapter. In algebra, the basic topics used, in order to deal with lattice theory, revolve around polynomials. Calculus is mainly about integral transforms, precisely, the Fourier transform, which serves as basis for the FWHT. Probability stands as the substance for sampling and the CLT. The terms sample and random variable (r.v.) are used commutably in this text, referring to a given set of observations. The number of observations in a sample is the sample size, here denoted as s.

2.1 Algebra

This section covers algebra-related topics used in this work, namely polynomial rings and lattices.

2.1.1 Polynomial rings

In abstract algebra, the algebraic structure called ring is a set R of finite or infinite elements, provided with the binary operations of addition and multiplication (SHOUP, 2008). Inputs and outputs of referred binary operations belong to the R ring, and all elements of a given ring should assume a common nature, e.g., integers, matrices, or polynomials. As for the binary operations, addition is always commutative, that is, for $a, b \in \mathbb{Z}$, then $a + b = b + a$, whereas multiplication might not be commutative in some cases. An instance of the referred multiplication constraint is that, considering two matrices A and B of same order, it is not possible to state AB produces the same result of BA. If the commutative property does apply to multiplication, as it does for integers and polynomials, then the ring is called a commutative ring, which is in the interest of this work, precisely as finite commutative rings of polynomials, provided with integer coefficients and identity elements both for addition and multiplication. Those rings can be represented as Equation 2.1.

$$R[x] = \{r_{s-1}x^{s-1} + r_{s-2}x^{s-2} + ... + r_1 x + r_0 \,|\, r_i,\ s \in \mathbb{Z}\} \tag{2.1}$$

Since a finite polynomial ring representation should somehow express its modulus, this can be done by means of specific mathematical notation in which the denominator is called an ideal generator, as in Equation 2.2.

$$R[x] = \frac{\mathbb{Z}[x]}{x^s - x - 1}. \tag{2.2}$$

The ring example represented by equation 2.2 has a modulus of $x^s - x - 1$.

Finally, there are five properties for polynomial rings to satisfy:

i) **identity for addition**

represented as 0_A;

ii) **identity for multiplication**

represented as 1_A;

iii) **associative multiplication**

$a \cdot (b \cdot c) = (a \cdot b) \cdot c$;

iv) **commutative multiplication**

$a \cdot b = b \cdot a$; and

v) **distributive multiplication**

$a \cdot (b \cdot c) = a \cdot b + a \cdot c.$

2.1.2 Lattice theory

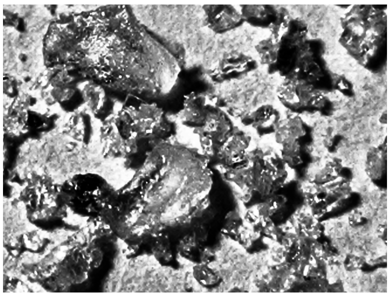
In this section, the subject known as lattice theory (GRÄTZER, 2011; WEHRUNG et al., 2016) is briefly covered. Such theory formalizes conceptual nets called lattices. A lattice is distinguished by its structural periodicity through which a pattern occurs. Mathematics can formalize that concept through many approaches, e.g., geometry, group theory, and order theory. Geometry has a less abstract approach, describing a lattice as an $\mathcal{L}$ geometric object, which is an array formed by i vertex or vector elements, as in $\mathcal{L}(v_1, v_2, \ldots, v_i)$ with $v_1, v_2, \ldots, v_i \in \mathbb{R}^j$, with $i,\ j \in \mathbb{N}$.

This versatile theory finds applications in many fields derived from chemistry and physics, like crystallography and communications (HEFFERON, 2017). In cryptology, lattice theory serves as a framework to answer a number of demands. A few of those are covered in section 2.5.

2.1.2.1 A concrete example

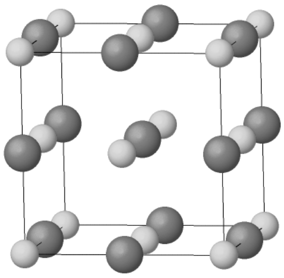
Although lattice theory is an abstract subject with a variety of approaches, it can be introduced by a concrete and simple instance. In chemistry, the table salt molecule makes an example of a real-world lattice. Figure 1 portrays an equalized greyscale photography of real table salt, and Figure 2 presents the basic structure of its molecule.

Figure 1: Photography of table salt.



Source: author (2021).

Figure 2: Lattice of a salt molecule.



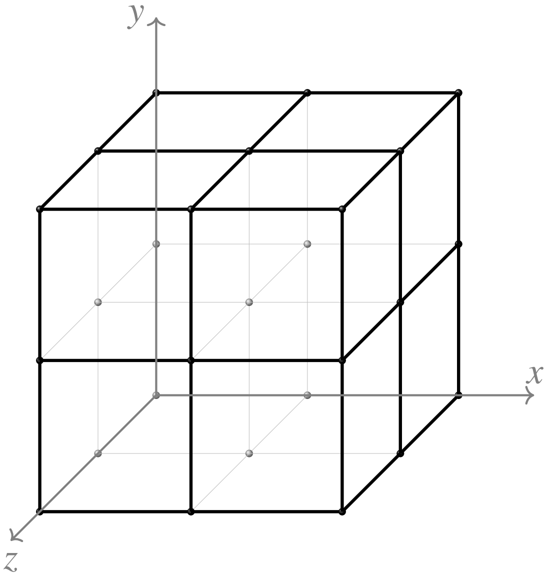
Source: author (2021).

The table salt in Figure 1 is a compound known as sodium chloride, whose molecule can be denoted by the three-dimensional crystal structure in Figure 2, with bigger darker atoms representing the chloride element and the smaller lighter atoms representing the sodium element. Still in Figure 2, considering the distances involved, element combination, and angles, a structural repetition can be observed in the cuboid parallelepiped, regarding atom order. By comparing the atom sequence in parallel edges, a repetition is observed, and that periodic arrangement constitutes the basic idea supporting lattice theory in mathematics. As the structure repeats itself, it is possible to infer bigger lattices by joining polyhedrons like that of Figure 2.

By removing atoms in the representation of Figure 2, and adding three-dimensional

Euclidean space reference axes, as in Figure 3, it is possible to visualize the existing lattice structure, thus providing geometric means to deal with the referred lattice. The lattice theory formalizes those concepts as to offer a tool to many areas of knowledge other than chemistry.

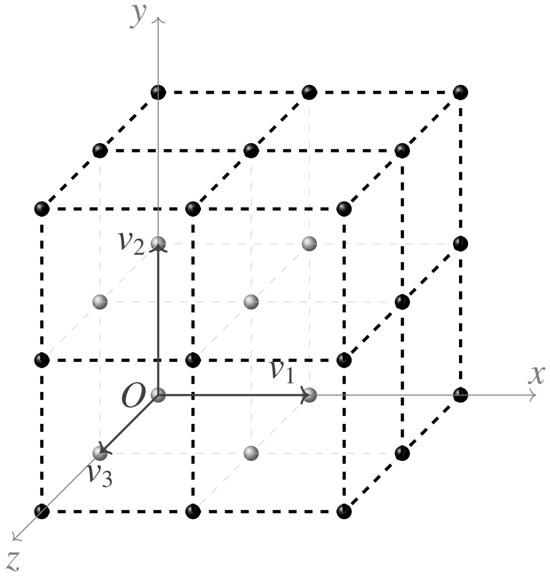
Figure 3: The lattice structure of a salt molecule.



Source: author (2021).

The observed cubic pattern in Figure 3 is supposed to replicate itself in any of the three axes. Lattice theory uses that idea to build on its pattern periodicity concepts for complex algebraic structures. In linear algebra, a linear combination can describe such structural recurrence through the unit vectors composing the basic parallelepiped. Since our example deals with a three-dimensional geometry, the basic polyhedron from Figure 2 is able to provide up to three linearly independent vectors. For algebraic generalization intentions, one might consider higher-dimensional polytopes, which allows i basis vectors, $i \in \mathbb{N}$, each of them featuring a j norm, $j \leq i, j \in \mathbb{N}$, which can be denoted as $v_1, v_2, ..., v_i \in \mathbb{R}^j$. Remarkably, as a generalization, i and j do not necessarily assume the same value. However, if the number of vectors i equals the norm j for each basis vector, the lattice is categorized as full-rank, which is the standard adopted in this work. The use of full-rank lattices for cryptographic purposes is covered in Section 2.5. For simplicity, considering a three-dimensional Euclidean space, as in Figure 3, it is possible to represent basis vectors $v_1, v_2, v_3 \in \mathbb{Z}^3$, presented by Figure 4. Such representation emphasizes the lattice elements arranged in the vertices, to the detriment of edges, which are now denoted by dashed lines. This is a lattice graph known as a grid.

Figure 4: A delimited region of a cubic lattice, and a possible basis.



Source: author (2021).

Figure 4 portrays a delimited region of a lattice in a Euclidean space originated in O, and its correspondent basis, which is orthogonal for this example. Infinite elements of that lattice may be obtained by combining v_1, v_2, and v_3. Although the example in Figure 4 portrays a lattice with an orthogonal basis, that might not always be the case. At that point, where bases are not orthogonal, lattice theory starts offering levels of computational complexity enough to make it interesting to cryptography. Later on, in Section 2.5, the lattice theory approach to cryptography is resumed. But before that, and in order to get to the referred section with the proper comprehension bases, it is also relevant to present the ring learning with errors (RLWE) problem, plus a few concepts on probability, and on the discrete Fourier transform (DFT).

2.2 The ring learning with errors problem

Firstly described in (LYUBASHEVSKY; PEIKERT; REGEV, 2013b; LYUBASHEVSKY; PEIKERT; REGEV, 2013a), the learning with errors over rings or ring learning with errors (RLWE) problem is a particular case of the learning with errors problem (LWE), which is a question over a system of equations, as of how random the referred equations are. LWE assumes equations may be artificially perturbed, then it takes them in pairs and tells which pairs are artificially random, and which are uniformly random. RLWE in its turn is an algebraically structured LWE, i.e., it implements LWE by means of polynomial rings, and in that sense, its origins are preceded by a structure the authors have previously adopted in the SWIFFT set of compression functions (LYUBASHEVSKY et al., 2008).

In practice, by enforcing the same security of LWE, RLWE is often more efficient, and if a public key is generated by RLWE instead of LWE, it occupies less memory.

There are two versions of RLWE, the decision version and the search version. Those versions are respectively based on the decision and search versions of the lattice problem known as the shortest vector problem (SVP). Decision RLWE is easier than search RLWE for the purpose of ideal lattices approximation, and for the matter of classical hardness characterization of the problem as a whole, hardness inherent to the decision version prevails. The decision version relies on figuring out pseudorandomness in a group of polynomials. More precisely, the goal in decision RLWE is to find out how pseudorandomness occurs in error distributions. The search version is about finding one polynomial, given a group of polynomials. More precisely, the goal in search RLWE is to recover a secret polynomial from the R_q polynomial ring (LYUBASHEVSKY; PEIKERT; REGEV, 2013b). For both versions, the ring has two moduli, $f(x)$ and q. Considering the even sample size output variable s (lowercase s), as described in 3.1, function $f(x) = x^s + 1,\ s \in \mathbb{Z}$, denotes a generic $R[x]$ polynomial ring modulus, as described in 2.1.1. The second modulus an RLWE polynomial ring should obey is q, a large public prime. The notation for such a ring is present in Equation 2.3.

$$R_q = \frac{\mathbb{Z}_q[x]}{x^s + 1}. \tag{2.3}$$

Security parameter s also implies polynomials in R_q should have degrees smaller than s (LYUBASHEVSKY; PEIKERT; REGEV, 2013b).

Thus, by considering a set of s monic polynomials as in:

$$\begin{aligned}
p_1(x) &= a_{1\,1}x^{s-1} + a_{1\,2}x^{s-2} + \ldots + a_{1\,s-2}x^2 + a_{1\,s-1}x + a_{1\,s} \\
p_2(x) &= a_{2\,1}x^{s-1} + a_{2\,2}x^{s-2} + \ldots + a_{2\,s-2}x^2 + a_{2\,s-1}x + a_{2\,s} \\
&\vdots \\
p_{s-2}(x) &= a_{s-2\,1}x^{s-1} + a_{s-2\,2}x^{s-2} + \ldots + a_{s-2\,s-2}x^2 + a_{s-2\,s-1}x + a_{s-2\,s} \\
p_{s-1}(x) &= a_{s-1\,1}x^{s-1} + a_{s-1\,2}x^{s-2} + \ldots + a_{s-1\,s-2}x^2 + a_{s-1\,s-1}x + a_{s-1\,s} \\
p_s(x) &= a_{s\,1}x^{s-1} + a_{s\,2}x^{s-2} + \ldots + a_{s\,s-2}x^2 + a_{s\,s-1}x + a_{s\,s}
\end{aligned}$$

The s polynomials constitute an R_q finite commutative ring of polynomials, as explained in Section 2.1.1. By making the referred polynomials equal to zero, and separating their constant terms, it is possible to write the following system of equations:

$$\begin{cases} a_{1\,1}x^{s-1} + a_{1\,2}x^{s-2} + \ldots + a_{1\,s-2}x^2 + a_{1\,s-1}x = -a_{1\,s} \\ a_{2\,1}x^{s-1} + a_{2\,2}x^{s-2} + \ldots + a_{2\,s-2}x^2 + a_{2\,s-1}x = -a_{2\,s} \\ \vdots \\ a_{s-2\,1}x^{s-1} + a_{s-2\,2}x^{s-2} + \ldots + a_{s-2\,s-2}x^2 + a_{s-2\,s-1}x = -a_{s-2\,s} \\ a_{s-1\,1}x^{s-1} + a_{s-1\,2}x^{s-2} + \ldots + a_{s-1\,s-2}x^2 + a_{s-1\,s-1}x = -a_{s-1\,s} \\ a_{s\,1}x^{s-1} + a_{s\,2}x^{s-2} + \ldots + a_{s\,s-2}x^2 + a_{s\,s-1}x = -a_{s\,s} \end{cases} \tag{2.4}$$

From that system, it is possible to write the following matrix equation:

$$\begin{bmatrix} a_{1\,1} & a_{1\,2} & \ldots & a_{1\,s-2} & a_{1\,s-1} & a_{1\,s} \\ a_{2\,1} & a_{2\,2} & \ldots & a_{2\,s-2} & a_{2\,s-1} & a_{2\,s} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ a_{s-2\,1} & a_{s-2\,2} & \ldots & a_{s-2\,s-2} & a_{s-2\,s-1} & a_{s-2\,s} \\ a_{s-1\,1} & a_{s-1\,2} & \ldots & a_{s-1\,s-2} & a_{s-1\,s-1} & a_{s-1\,s} \\ a_{s\,1} & a_{s\,2} & \ldots & a_{s\,s-2} & a_{s\,s-1} & a_{s\,s} \end{bmatrix} \times \begin{bmatrix} x^{s-1} \\ x^{s-2} \\ \vdots \\ x^2 \\ x \\ 1 \end{bmatrix} = \begin{bmatrix} -a_{1\,s} \\ -a_{2\,s} \\ \vdots \\ -a_{s-2\,s} \\ -a_{s-1\,s} \\ -a_{s\,s} \end{bmatrix}. \tag{2.5}$$

By representing the constant terms vector as b, Equation 2.5 can be written as:

$$A \times X = b. \tag{2.6}$$

Once matrix A is invertible, there is:

$$X = A^{-1} \times b. \tag{2.7}$$

In Equation 2.7, A^{-1} is the inverse matrix of A, the solution being represented by the X vector.

As previously stated, LWE problems are supposed to deal with artificially perturbed equations. Thus, Equation 2.7 is modified, and a pseudo-random error is added to it, hardening access to the X system solution:

$$X + e \neq A^{-1} \times b. \tag{2.8}$$

At this point, the problem is about distinguishing e influence in ring elements, i.e., in the left-hand side of Inequation 2.8. Tuning e is actually a relevant aspect for RLWE applications, and it should consider the large public prime q and the lattice dimension (LYUBASHEVSKY; PEIKERT; REGEV, 2013a).

2.3 Probability

This section covers discrete Gaussian sampling, probability distribution, the central limit theorem (CLT), and statistical metrics used herein. The terms Gaussian distribution and normal distribution are used interchangeably throughout this document.

2.3.1 Distribution

A distribution of probability can present itself in many sorts, and in any of those, it can be described by two mathematical functions. In the continuous case, they are the probability density function (PDF) and the cumulative density function (CDF). The latter keeps its name in the discrete case, but the former is called probability mass function (PMF).

A uniform distribution is a sort of probability distribution whose domain of possible samples offer zero probability, except for a bounded interval in which probability is a non-null positive constant. Binomial distributions are well-known discrete-case probability distributions, used for modelling success chances in yes/no experiments. Normal distributions, also called Gaussian distributions, are the ones used throughout the present work. They are explained in Section 2.3.1.1.

2.3.1.1 Normal distribution

Considering N as a one-dimensional independent random variable (BERRY, 1940) associated to a continuous-case normal distribution of probabilities, two parameters are typically used to describe it. The mean μ and the variance σ^2 are those parameters, as in equations 2.9 and 2.10.

$$N \sim \mathcal{N}(\mu, \sigma^2) \tag{2.9}$$

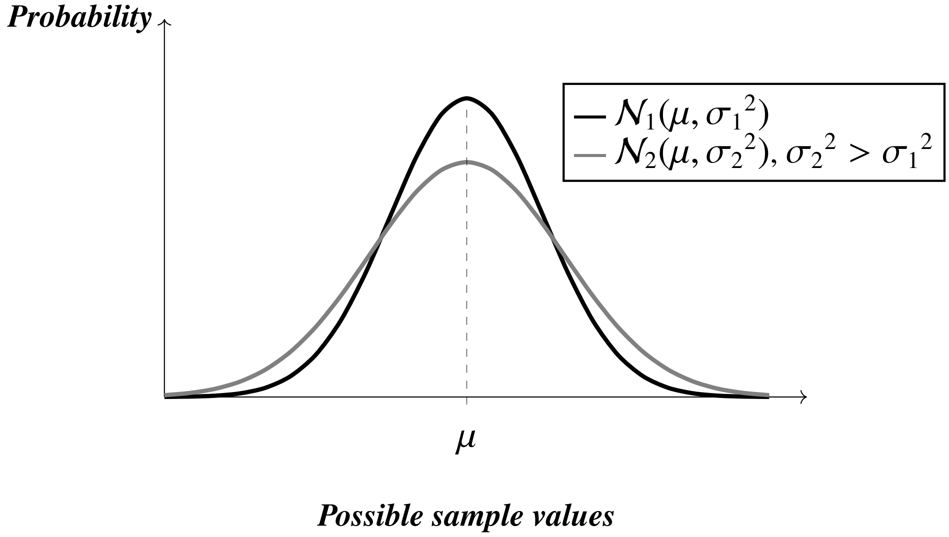
Equation 2.9 shows a simplified normal PDF representation. According to (WEISSTEIN, 2021), the function itself can be written as in Equation 2.10.

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot \mathrm{e}^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2},\ x \in \mathbb{R} \tag{2.10}$$

Thus, the N random variable denoted by Equation 2.9 corresponds to the probability function $P(x)$ expressed by Equation 2.10. In Figure 5, the graphical representation of two generic normal distribution PDF functions are portrayed. A continuous line and a dashed one representing respectively $\mathcal{N}_1$ and $\mathcal{N}_2$. They have the same generic mean value μ but different variance values ${\sigma_1}^2$ and ${\sigma_2}^2$, such that ${\sigma_1}^2 < {\sigma_2}^2$. Also, the tail-cut factor τ is the same for both.

Figure 5 reveals that bigger variance values spread the probability distribution more, that is, the chances of sampling values more distant from the mean increase while the chances of sampling values near the mean decrease. A numeric example is

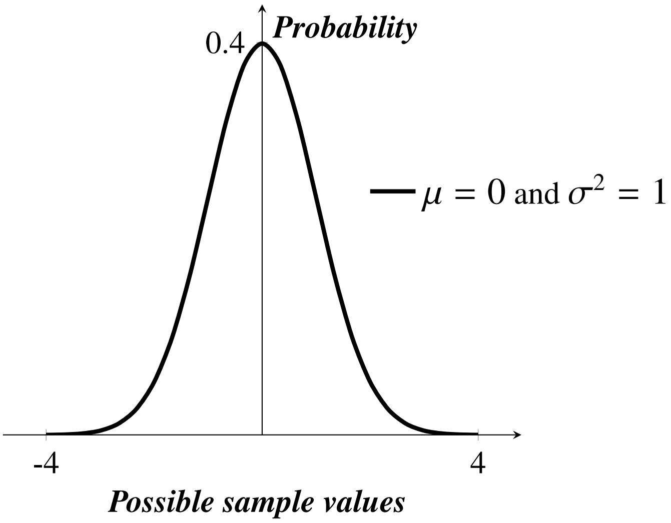
Figure 5: Two normal distribution PDF plots with different variance values.



Source: author (2021).

given in Figure 6. It portrays a normal distribution PDF graphical representation, in which the PDF is truncated or bounded inside the $[-4,\ 4]$ domain. Its mean is 0, and variance is 1, such values denoting a standard normal distribution.

Figure 6: A $[-4,\ 4]$ truncated-domain view of a standard normal distribution PDF.



Source: author (2021).

Though, in order to help in Gaussian PDF manipulation, this work enforces a third parameter, known as the tail-cut factor τ. It constitutes an artificial resource used in some works of lattice-based cryptography (KARMAKAR et al., 2018; ZHAO; STE-

INFELD; SAKZAD, 2020), serving the purpose of interval manipulation. Gaussian domain then becomes $[\mu - \tau\sigma, \mu + \tau\sigma]$.

Lastly, as for sampling uncertainty, if proper mean and variance values are used, normal distributions offer Shannon entropy values at least as great as those of uniform distributions (COVER; THOMAS, 2006; MORAIS, 2018).

2.3.2 Sampling

According to (HOUSE, 1989), sampling is the process of selecting a sample for a given purpose, e.g., analysis. A sample may be composed by one or more observations. In probability and statistics, sampling is a standard procedure performed on probability distributions. In cryptology, it supports the concept of random value generation.

2.3.2.1 Discrete Gaussian sampling

Gaussian sampling refers to sampling from a normal or Gaussian probability distribution. Discrete Gaussian sampling (DGS) is a term used to indicate a Gaussian sampling of integer values. In lattice-based cryptography, it stands as a primitive for random number generation.

The sampling precision required by cryptography represents computationally costly routines. In general, DGS complexity resides in low-level mathematical evaluations of integrals and series, which require specialist routines (SAARINEN, 2015).

As a pillar of lattice-based cryptosystems (GENISE et al., 2020), DGS has received many recent contributions, offering a variety of methods, such as (KARMAKAR et al., 2018; WANG, 2019; WANG; LING, 2019). Additionally, there are valuable works on computer-based Gaussian sampling in previous decades, e.g., (RADER, 1969).

In this work, the referred primitive picks its observations from a truncated normal distribution probability mass function (PMF). It is strenghtened by a shuffling strategy based on the fast Walsh–Hadamard transform (FWHT). Often in lattice-based cryptography, the DGS primitive is used to build polynomial rings, but here it generates noise for already existing polynomials, according to RLWE premises.

2.3.3 Sampling-related algorithms

As the random number generation primitive of this work is a discrete Gaussian sampler, this section portrays some notorious sampling strategies.

2.3.3.1 Algorithm: rejection

The rejection sampling strategy follows the premise that computations not achieving certain criteria should be discarded. Its first formal explanation supposedly happens in (NEUMANN, 1951), as the author discusses the production of randomness by physical methods, through coin tossing. By assuming independence of two successive tosses, he proposes that at least one pair of tosses is necessary for obtaining one single random result, which can be either heads or tails. In such process, two successive tosses are made. Then if the results are equal, those two tosses are rejected, and another pair of tosses takes place, until the results are different. Thus, it is emphasized that potential process repetition stands as a downside for rejection sampling, which in its turn, can be mitigated by means of having random values recorded somewhere.

2.3.3.2 Algorithm: binary

This section covers a sampling strategy for discrete distributions known as binary, which was introduced by (DUCAS et al., 2013) for the lattice-based BLISS digital signature scheme. The binary sampling algorithm builds upon a combination of two strategies, inversion sampling and rejection sampling (NEUMANN, 1951; DWARAKANATH; GALBRAITH, 2014). While the inversion strategy translates sampling from a normal distribution into a uniform one on a different set, the rejection strategy samples from a slightly modified normal distribution, and then performs a rejection test on the result.

In BLISS, rejection sampling uses two resembling, but still distinct, normal distributions. They are mixed, and the result is a single bimodal distribution, whose PDF is presented by Equation 2.11.

$$P(x) = 2^{-x^2},\ x \in \mathbb{Z} \tag{2.11}$$

Regarding the rejection test, Bernoulli functions are used, here denoted as $\mathcal{B}(x)$. BLISS expects rejection tests to return either zero or one. It needs no precomputed table, relying on a binary representation system supposed to compensate probability distributions being computed on the fly (FOLLÁTH, 2014).

A possible binary sampling construction is presented in Algorithm 2.1, which is a contribution of this work.

Algorithm 2.1 A possible binary sampling construction.

```
algorithm binary_dgs ( $k$ : natural)

    $y \leftarrow$ random from $\{0,\ 1,\ ...,\ k-1\}$;

    do
        $x \leftarrow$ random from $P(x)$;
    while ($\mathcal{B}(x) = 0$)

    return $kx + y$;

end algorithm
```

Source: author (2022).

In Algorithm 2.1, variable y stores the inversion sampling result, and variable x

stores the rejection sampling result. Line number seven represents the Bernoulli function performing the rejection test for the random value stored in x. If the Bernoulli rejection test returns a zero, then the value stored in variable x is rejected, and the rejection sampling routine is performed again, followed by another rejection test. That routine repeats until the Bernoulli function returns one, meaning the value in x is accepted. Finally, the algorithm returns the result of $kx + y$. That is a simple explanation of binary sampling. There are modified approaches, such as the one present in the FACCT sampler (ZHAO; STEINFELD; SAKZAD, 2020).

2.3.3.3 Algorithm: cumulative distribution table

This section presents a DGS algorithm built upon a cumulative distribution function, namely the cumulative distribution table (CDT) sampling algorithm. Besides denoting the name of the algorithm, CDT also refers to the table used by that algorithm. Before sampling, the cumulative distribution function must fill that table with probability values. Thus, in CDT, persistence must precede sampling, i.e., a regular CDT strategy should consider the implications of memory access and memory consumption.

The core of the algorithm takes a random real value $r \in [0, 1)$, and performs a search operation on the precomputed table, comparing r to each value of the table. The last value of the CDT to be bigger than r has its index s returned as a valid observation for the sample. More formally, the algorithm tries to find an s index such that the CDT $_{s-1} < r <$ CDT $_{s}$ condition is satisfied. According to (ZHAO; STEINFELD; SAKZAD, 2020), the CDT-based DGS can be $O(\tau\sigma)$, its main drawback being the high memory consumption involved, as the algorithm requires a precomputed CDT.

DGS routines like that are specially relevant to this work for their generic construction is used as a basis for comparisons with the FWHT strategy. The use of a CDT-based DGS algorithm as reference is justified by the fact many lattice-based cryptography works use it for Gaussian sampling, as mentioned in Section 1.4. One example is (ALKIM et al., 2019), which presents a detailed CDT-based algorithm, which includes the table filling. Their construction receives a seed and a nonce as inputs, and then it outputs a sequence of samples. The isochronous construction uses a precision variable, and keeps track of the original sampling order. Another example is (HOWE et al., 2019), which presents a Gaussian sampler based both on CDT and rejection. Just as in (ALKIM et al., 2019), this construction is isochronous, and it also enforces a precision variable to both generate cumulative distribution tables, and get random-like numbers in $[0, 1)$. The latter are then used to obtain index values from the cumulative distribution table initially generated. More specifically, after getting a random number from $[0, 1)$, that number is compared to each value of the table. Then, the last value of the table to be greater than the referred number has its index chosen as a sample observation.

A possible CDT sampler, based upon (ALKIM et al., 2019) and (HOWE et al., 2019), is presented by Algorithm 2.2, which is a contribution of this work. It works in the $[0, 1]$ interval, and its input parameters are a cumulative distribution table, named cdt, and the desired number of observations to compose a sample. The natural value in n holds the number of elements in the table, that is, the number of elements to be in the cdt input variable. The n_aux variable serves as an iteration control through the table. Variable $nobs\_aux$ is used to control iterations through v. Variable p stands for the natural-valued precision used to populate the table. Variable r is a random value, which can also be used as a seed, depending on the sample size. It is uniformly obtained by means of the $generate\_random()$ function sampling from the $[0, 1]$ real interval. The integer vector v stores index values obtained, and it is returned by the end. According to (HOWE et al., 2019), isochrony enforcement is achieved if each r value takes the entire table to be read.

Before using a CDT-based sampler, it is necessary to compute a cumulative distribution table, using a precision of p .

Algorithm 2.2 A possible CDT-based DGS construction.

```
algorithm cdt_dgs ($cdt$ : vector, $nobs$ : natural)
    $n$ :          natural;
    $n\_aux$ :     natural;
    $nobs\_aux$ :  natural;
    $p$ :          natural;
    $r$ :          real;
    $v$ :          integer vector;

    $n \leftarrow get\_length(cdt)$ ;
    $p \leftarrow get\_precision(cdt)$ ;
    $v \leftarrow 0$ ;

    for ( $nobs\_aux$ from $0$ to $nobs - 1$ , step $1$ )
        $r \leftarrow generate\_random([0, 1))$ ;
        for ( $n\_aux$ from $0$ to $n - 1$, step $1$ )
            if $r < cdt_{n\_aux}$
                $v_{nobs\_aux} \leftarrow n\_aux$ ;
                break;
            end if
        end for
    end for

    return $v$;
end algorithm
```

Source: author (2022).

The returned variable v stands for a sample.

2.3.3.4 Algorithm: Knuth–Yao

This section presents an overview of the Knuth–Yao sampling algorithm (KNUTH; YAO, 1976). In the article, a known non-uniform probability distribution is sampled via random walk in a type of binary tree called a discrete distribution generating (DDG) tree. Authors end up concluding that further investigation is needed as to understand how efficiently a normal distribution can be generated with simple algorithms. According to the text, the ability to work with binary representations could be a desirable characteristic for such algorithms. The referred DDG tree can be implemented as a matrix, according to the following rule: the number of nodes in the tree at the ith level should be equal to the Hamming weight of the matrix in its ith column. Each leaf node corresponds then to a sample, denoted here as z.

A possible construction is presented in Algorithm 2.3, which is a contribution of this work. It receives a probability matrix v as an argument, and returns a sample z. Variable d is the distance between the visited node and the rightmost internal node of the DDG tree. Variable t should be zero until sampling hits a terminal node, which changes its value to one.

Algorithm 2.3 A possible Knuth–Yao DGS construction.

```
algorithm dgs_knuthyao ($v$: real matrix)
    $d \leftarrow 0;$
    $j \leftarrow 0;$
    $t \leftarrow 0;$

    do
        $d \leftarrow 2d + random\_bit();$

        for ($i \leftarrow max\_row;\ i > 1;\ i--$)
            $d \leftarrow d - v_{i\,j}\ ;$

            if $d = -1$ then
                $z \leftarrow i;$
                $t \leftarrow 1;$
                break;
            end if
        end for

        $j{+}{+};$
    while $t = 0$

    return $z;$

end algorithm
```

Source: author (2022).

Implementations of the original Knuth–Yao sampling strategy used to experience two main downsides, inefficiency and potential information leakage. The former refers to the use of arbitrary-precision in floating-point arithmetic, and the latter is related to timing vulnerabilities. Recent versions of Knuth–Yao are proposed in (ROY; VERCAUTEREN; VERBAUWHEDE, 2014; KARMAKAR et al., 2018; MICCIANCIO; WALTER, 2018). They deal with the referred downsides using different approaches.

In (KARMAKAR et al., 2018), authors highlight the referred timing vulnerabilities in the original algorithm, presenting then an isochronous construction as to mitigate those risks. Although it does not require large precomputed tables, its routines execution demands considerable memory.

2.3.4 Cental limit theorem

The central limit theorem (CLT) refers to a number of enunciations, some of them being referenced in Section 1.4. The core idea behind most enunciations is that by sampling randomly and repeatedly from a given population distribution, e.g., a uniform distribution, and then computing an average for each sample, the resulting distribution of averages tends to a normal distribution. Samples correspond to any uncorrelated random variables, and the obtained distribution of averages tends to a normal random variable (RADER, 1969; MENDONÇA, 2022). Thus, the CLT can also be used to estimate values, e.g., as a maximum likelihood estimator (MLE).

In this work, the CLT is verified, and most importantly, it is exploited in order to accelerate the generation of Gaussian random variables, e.g., seemingly normal distributions, in a more efficient fashion. The $\mu_{\mathfrak{D}}$ mean of 2^{β} random variables approaches the theoretical population average as the number of samples grows. Equation 3.3 presents the formula used to obtain $\mu_{\mathfrak{D}}$. As mentioned, a supposedly uniform population distribution is to be sampled. In this point lies a part of the CLT exploitation, precisely, in population choice. For that matter, the chosen population is previously manipulated, forcing the original, supposedly uniform population distribution to be normal already. Considering the samples obtained in this work, it is desirable to understand how far results are from a trustable normal distribution.

2.3.5 Metrics

This section briefly presents the metrics chosen to be used by this work. Further information, e.g., adapted formulas, using the variables of this work, are supplied by Section 3.1.

Shannon entropy: it constitutes a metric to assess the uncertainty of random variables (COVER; THOMAS, 2006). According to (KARMAKAR et al., 2018), higher values of standard deviation produce higher values of entropy.

Relative entropy: it is also known as the Kullback–Leibler divergence (COVER; THOMAS, 2006). The relative entropy is a particular case of the Rényi divergence. It is used to obtain the amount of information one probability distribution has of another probability distribution.

Mean: a metric of the normal distribution PMF functions, which is expected to be near zero, because that is the fixed mean value of the algebraic Gaussian.

Standard deviation: measure of dispersion, determining a confidence interval to sample from.

Coefficient of skewness: it regards the symmetry of the curve, corresponding to the third moment of the discrete Gaussian.

Kurtosis: a measure of quality, corresponding to the fourth moment of the discrete Gaussian.

2.4 Discrete Fourier transform

The Fourier transform is a mathematical function of frequency, often represented as $g(\omega)$, used to find the frequencies of a time-domain function, often $f(t)$. For an integrable function $f(t)$, it is possible to write $g(\omega)$ as in Equation 2.12:

$$g(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i2\pi\omega t}\, dt,\ \omega \in \mathbb{R}. \tag{2.12}$$

In equation 2.12, $g(\omega)$ is a complex value, and it tells how much the ω frequency is present in $f(t)$. As a function of frequency, the Fourier transform has conceptual ties with PDF functions, because both frequency and probability carry the idea of likelihood (GILLIES, 2012). In fact, relative frequency is closely related with probability, e.g., it is possible to carry out a probability experiment with a given range of possible results. After a number of repetitions, by observing the relative frequencies for the referred results, these values approach the ones previously obtained with probability (NEYMAN, 1937). The Fourier transform of a PDF, e.g., $P(x)$, actually results in a characteristic function, which can be used as an alternative probability distribution analysis means (SAKAMOTO; MORI; SEKIOKA, 1997; KARDAR, 2019), as in Equation 2.13. In this case, it is necessary to treat the probability domain of possible sample values as a time domain.

$$g(\omega) = \int_{-\infty}^{\infty} P(x)e^{-i2\pi\omega x}\, dx,\ \omega \in \mathbb{R}. \tag{2.13}$$

The characteristic function expressed by Equation 2.13 basically denotes the expected value for the random variable N (see Equation 2.9). As the expected value of a given random variable returns an average value, thus it is possible to infer Equation 2.13 results in μ.

In terms of computational complexity, the Fourier transform integral covers the full real line, which makes it unattractive, mostly if efficiency is regarded. However, its approximation known as discrete Fourier transform (DFT) rises as a computationally-viable option (ORSINI, 1994). Generic formulation of the DFT can be put as that of equation 2.14, which operates on a finite number of n samples:

$$g(\omega) = \sum_{t=0}^{n-1} f(t)e^{-i2\pi\omega t}, \ \omega \in \mathbb{R}. \tag{2.14}$$

As in equation 2.12, in equation 2.14, the $g(\omega)$ value is a complex number showing how much the ω frequency is present in $f(t)$, but this time regarding only the considered sequence of samples. Algorithms to implement the DFT are typically $O(n \log n)$, like (COOLEY; TUKEY, 1965). Called fast Fourier transform (FFT) algorithms, they are used in many applications, e.g., image processing. As enhancements kept coming for FFT algorithms, the use of Hadamard matrices stood out as a valuable feature (PRATT; KANE; ANDREWS, 1969).

In this work, for the benefit of efficient Gaussian sampling, an FFT algorithm known as the Fast Walsh–Hadamard transform (FWHT), based on Hadamard matrices, is used. The FWHT is introduced in Section 2.4.1.3.

2.4.1 The Hadamard matrix and transform

An Hadamard matrix H can be described as a symmetric, and thus square, matrix whose elements are either one or minus one, meaning H obeys a binary system. Not all orders are possible for the construction of such matrices, e.g., it is impossible to build an Hadamard matrix of order three (RADER, 1969). Though, it is relatively simple to build it with order 2^β, $\beta \in \mathbb{N}^*$. Examples are provided in equations 2.15, 2.16, and 2.17.

$$H_{\beta=1} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.15}$$

$$H_{\beta=2} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \tag{2.16}$$

$$H_{\beta=2} = \begin{bmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{bmatrix} \tag{2.17}$$

From 2.16, and 2.17, it can be seen there might be more than one Hadamard matrix for a given 2^{β} order. However, some works, like (RADER, 1969), only make use of those in which both the initial row and column are composed of plus ones.

As for notation, considering an Hadamard matrix H of order h, H^T denotes its transpose, and I_h denotes an identity matrix of order h. Some properties are presented next.

Property one: since H is symmetric, that is, $h_{i,j} = h_{j,i}$, its transpose H^T is also an Hadamard matrix.

Property two: for matrices of order h, $H \times H^T = h \times I_h$.

Property three: $H \times H^T = H^T \times H$.

Property four: orthogonality is verified for rows, and also for columns, of H.

Property five: from property four, by comparing two rows or two columns of H, half the elements in one vector will find correspondence in the other.

Property six: known as the Sylvester method, it says that if H is an Hadamard matrix of order 2^β, then $\begin{bmatrix} H & H \\ H & -H \end{bmatrix}$ is also an Hadamard matrix (SYLVESTER, 1867). The sixth property is used to generate the matrix in equation 2.16 from the one in equation 2.15. As in (RADER, 1969), for both matrices, an element $h_{i\ j}$ can be described as in Equation 2.18.

$$h_{i\ j} = (-1)^{h_{1\ 1}}\,(-1)^{h_{2\ 2}} \ldots (-1)^{h_{2^\beta\ 2^\beta}} \tag{2.18}$$

Property seven: considering an Hadamard matrix in which both the initial row and column are formed by plus ones, i.e., elements $h_{i\ j}$, $h_{1\ j} = 1$ and $h_{i\ 1} = 1$, then there are two rows which, by having their elements multiplied produce a third row which is also in the matrix (RADER, 1969), as in Equation 2.19. The cited work also points out that such property means $h_{l\ j}$ is the result of an $\oplus$ (exclusive or) operation between elements of rows i and k, and that, by symmetry, it holds true for columns.

$$h_{i\ j} \cdot h_{k\ j} = h_{l\ j} \tag{2.19}$$

The Hadamard transform is a DFT in $\mathbb{R}$, basically constituting an operation of multiplication between two matrices. One of those matrices is an H matrix, as in equation 2.20, which portrays an Hadamard transform of a given matrix η. In its turn, matrix η is a 2^β-lengthened initialization vector, storing randomly-generated binary values obtained from the normal distribution.

$$g(\omega) = H \times \eta \tag{2.20}$$

2.4.1.1 Randomness in the Hadamard matrix

H is said to be normalized if the first row and column contain only ones (HARWIT; SLOANE, 1979), e.g., the matrices in equations 2.15 and 2.16. By removing the referred row and column, a square matrix G of order $h - 1$ is obtained. Now if ones are changed to zeros, and minus ones to ones, then an S-matrix of order $h - 1$, S_{h-1}, is produced. For the 2.16 equation, its referring S-matrix is described in equation 2.21.

$$S_3 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} \tag{2.21}$$

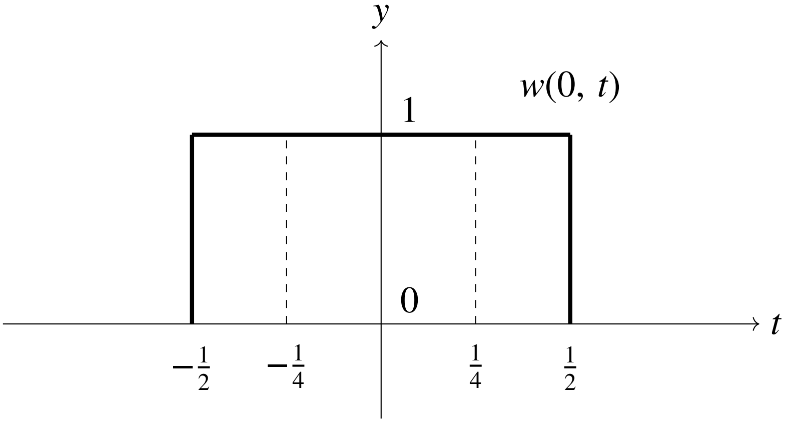
If a given S-matrix has the property of having each row as a left shift of the former, as in equation 2.21, then that S-matrix is said to be cyclic, and its first row is considered a pseudo-random sequence (HARWIT; SLOANE, 1979).

2.4.1.2 Relationship with Walsh functions

Walsh functions describe zero-centered square waves in the $-\frac{1}{2} \leq t \leq \frac{1}{2}$ domain, and their general form can be expressed as $w(i, t)$, $i \in \mathbb{N}$, i being the number of times the t axis is crossed. So, $w(6, t)$ crosses the t axis six times and is called the 6th Walsh function. Similarly, $w(7, t)$ crosses the t axis seven times and is called the 7th Walsh function, and so on. According to (HARWIT; SLOANE, 1979), the most important property of Walsh functions is that they form a complete orthonormal family of functions.

Hadamard matrices relate to Walsh functions because, considering the former, alternation of values in a given row can be treated as a function whose behavior is similar to that of the latter (SYLVESTER, 1867; HARWIT; SLOANE, 1979; BEER, 1981; EVANGELARAS; KOUKOUVINOS; SEBERRY, 2003). The first row of matrix $H_{\beta=2}$ in equation 2.16 is $1,\ 1,\ 1,\ 1$. Starting in the first element, it is possible to verify there is zero changes of sign from a position to its next. So, the referred Hadamard matrix associates to the 0th Walsh function, or $w(0,\ t)$, because the latter expresses what happens in the transitions between the Hadamard matrix row elements. In Figure 7, the Walsh function to describe transitions taking place in the first row of $H_{\beta=2}$ is presented.

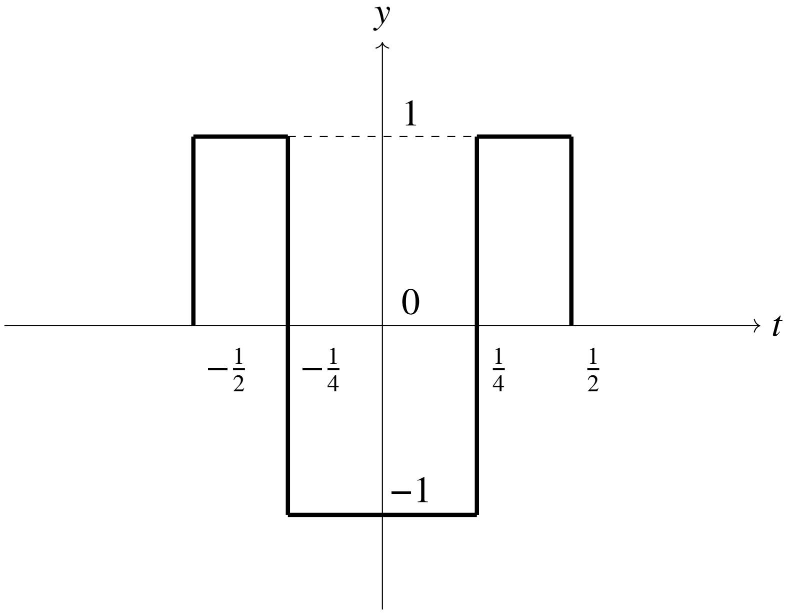
Figure 7: The 0th Walsh function.



Source: author (2021).

An additional example considers a row formed by $1,\ -1,\ -1,\ 1$ from another $H_{\beta=2}$ matrix. Starting in the first element, it is possible to verify two changes of sign, the first occuring in the transition from element one to element two, and the second occurring in the last transition. It is an association to the 2nd Walsh function, or $w(2,\ t)$, which is presented in Figure 8.

Figure 8: The 2nd Walsh function.



Source: author (2021).

Axiom 1. *From (HARWIT; SLOANE, 1979), it is possible to infer that if a given Hadamard matrix row relates to the i th Walsh function, then all Walsh functions to cross the t axis a number of times smaller than i also relate to the referred Hadamard matrix.*

2.4.1.3 The Fast Walsh–Hadamard transform

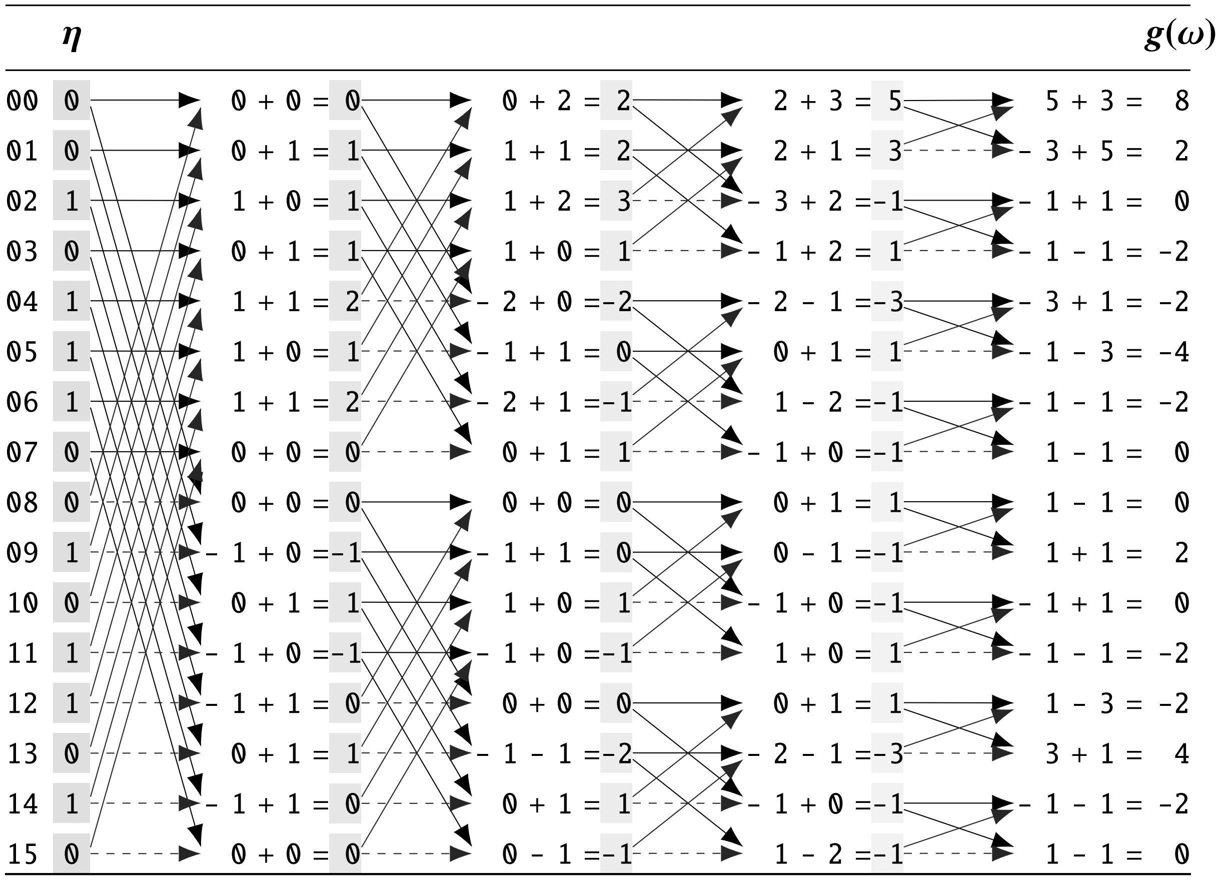
Hadamard transforms can benefit from the fact Hadamard matrices are populated by ones and minus ones elements only. Thus, the matrix product, represented by $g(\omega)$ in equation 2.20, can be obtained without multiplication operations between scalars, only additions being used (EVANGELARAS; KOUKOUVINOS; SEBERRY, 2003). Remarkably, that characteristic meets a suggestion in the conclusion of (KNUTH; YAO, 1976), as explained in 2.3.3.4. Done that way, the transform is said to be fast, and it is called a fast Hadamard transform, or a fast Walsh–Hadamard transform (FWHT). The latter emphasizes the relationship between Hadamard matrices and Walsh functions, something that is explained in Section 2.4.1.2.

The number of steps needed to compute an Hadamard transform of order n, $H_{\beta=log_2 n}$ is $n^2 - n$, whilst a fast Hadamard transform reduces the number of steps to approximately $n \log_2 n$ (HARWIT; SLOANE, 1979), or simply $n\beta$, considering the main input variable of this work. Indeed, as mentioned in Section 1.4, speedups are reported by some works using the FWHT. Besides benefiting from reduced computational complexity, adoption of FWHT still offers the randomness facilities of Hadamard matrices, as seen in Section 2.4.1.1. Though, only Sylvester method-supported Hadamard matrices, e.g., property six in Section 2.4.1, offer the referred fast transform option (HARWIT; SLOANE, 1979). This work uses those benefits to formulate an FWHT-based shuffling routine inside its discrete Gaussian sampler.

As an example, the diagram in Figure 9 presents the transform evolution of an initialization vector (IV) with sixteen positions, that is, $\beta = 4$. Loaded with binary values, the vector corresponds to the η matrix of Equation 2.20, and the resulting FWHT of η is presented in column $g(\omega)$. Continuous arrows generate positive numbers, and dashed arrows generate negative numbers. A stage can be understood as a vector shuffling procedure. There are $\beta = 4$ stages, and in each stage, 2^4 operations are computed, one for each vector position.

Regarding the FWHT inverse computation, a number of β stages corresponds to solving β systems of 2^β equations in a set of 2^β variables each.

Figure 9: Diagram of FWHT for input vector size 16.



	η				$g(\omega)$
00	0	0 + 0 = 0	0 + 2 = 2	2 + 3 = 5	5 + 3 = 8
01	0	0 + 1 = 1	1 + 1 = 2	2 + 1 = 3	- 3 + 5 = 2
02	1	1 + 0 = 1	1 + 2 = 3	- 3 + 2 =-1	- 1 + 1 = 0
03	0	0 + 1 = 1	1 + 0 = 1	- 1 + 2 = 1	- 1 - 1 = -2
04	1	1 + 1 = 2	- 2 + 0 =-2	- 2 - 1 =-3	- 3 + 1 = -2
05	1	1 + 0 = 1	- 1 + 1 = 0	0 + 1 = 1	- 1 - 3 = -4
06	1	1 + 1 = 2	- 2 + 1 =-1	1 - 2 =-1	- 1 - 1 = -2
07	0	0 + 0 = 0	0 + 1 = 1	- 1 + 0 =-1	1 - 1 = 0
08	0	0 + 0 = 0	0 + 0 = 0	0 + 1 = 1	1 - 1 = 0
09	1	- 1 + 0 =-1	- 1 + 1 = 0	0 - 1 =-1	1 + 1 = 2
10	0	0 + 1 = 1	1 + 0 = 1	- 1 + 0 =-1	- 1 + 1 = 0
11	1	- 1 + 0 =-1	- 1 + 0 =-1	1 + 0 = 1	- 1 - 1 = -2
12	1	- 1 + 1 = 0	0 + 0 = 0	0 + 1 = 1	1 - 3 = -2
13	0	0 + 1 = 1	- 1 - 1 =-2	- 2 - 1 =-3	3 + 1 = 4
14	1	- 1 + 1 = 0	0 + 1 = 1	- 1 + 0 =-1	- 1 - 1 = -2
15	0	0 + 0 = 0	0 - 1 =-1	1 - 2 =-1	1 - 1 = 0

Source: author (2021).

The leftmost grayed column in Figure 9 presents the η initialization vector of Equation 2.22, storing binary values.

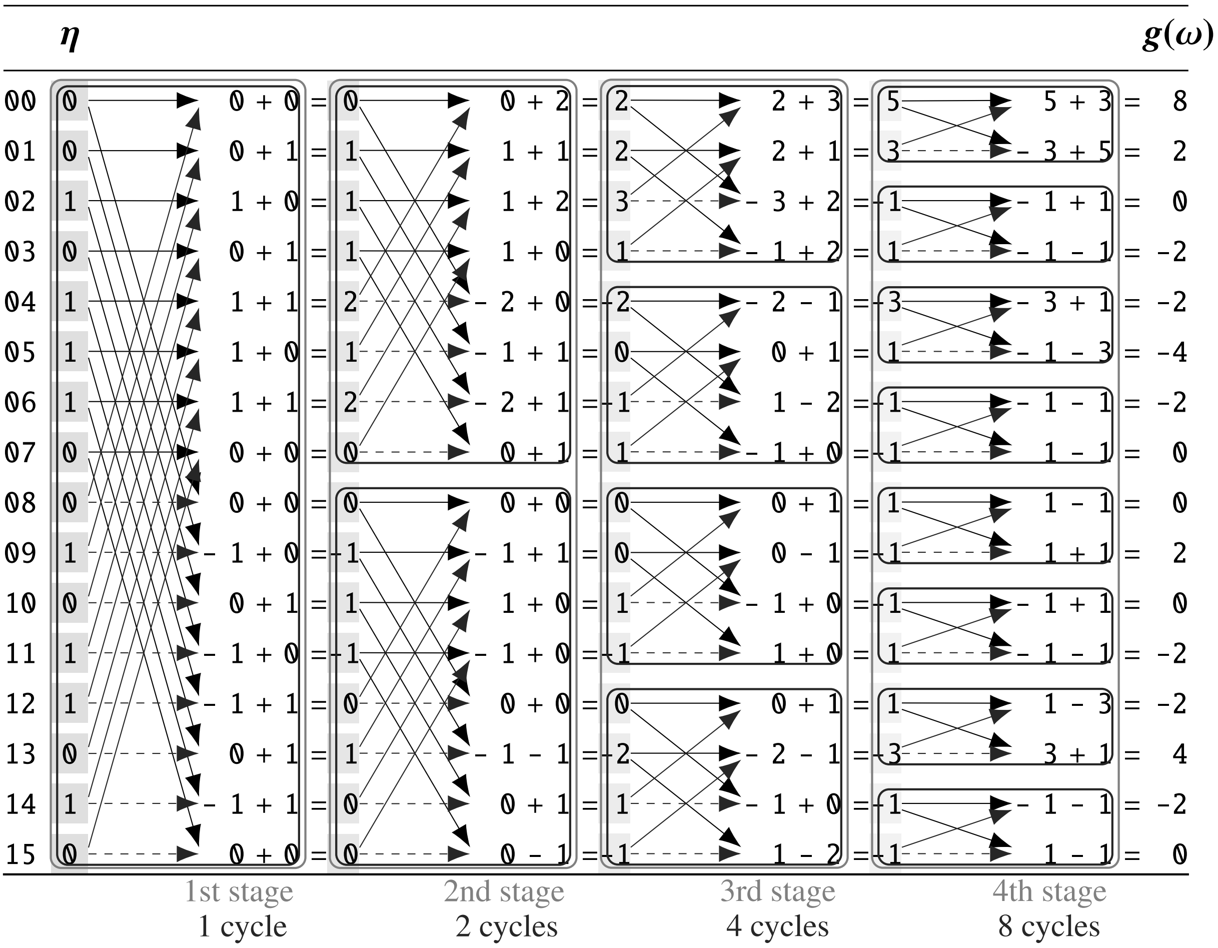
$$\eta = (0,\ 0,\ 1,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 1,\ 0,\ 1,\ 1,\ 0,\ 1,\ 0) \qquad (2.22)$$

As stages are done from left to right, columns of Figure 9 assume lighter tones of gray, indicating the transform computation is closer to its end, when the $g(\omega)$ vector is ready, as in Equation 2.23. Each $g(\omega)$ result consists in one observation for a sample being formed.

$$g(\omega) = (8,\ 2,\ 0, -2, -2, -4, -2,\ 0,\ 0,\ 2,\ 0, -2, -2,\ 4, -2,\ 0) \qquad (2.23)$$

Each stage contains one or more cycles. For convenience, in Figure 10, stages and cycles are contoured by approximate rectangles with rounded corners. It repeats Figure 9, this time with stages marked as four grayed rectangles, and cycles marked as inner darker gray rectangles. From left to right, the four gray rectangles correspond to the first, second, third, and fourth stages.

Figure 10: Diagram of FWHT with stages and cycles marked.



Source: author (2021).

Further details to support this work are presented in Section 3.2.

2.5 Lattice-based cryptography

This section intends to retrieve the concepts of lattice theory, in order to present their use within cryptography, in what is referred to as lattice-based cryptography. Earlier, in the introductory chapter, and then in Section 2.1.2, lattice theory was briefly presented. It constitutes a pillar for this section, since lattice-based cryptography is built upon nondeterministic polynomial (NP) problems from lattice theory. Lattice-based cryptosystems rely on the NP-hardness of those problems, e.g., the shortest vector problem (SVP), the closest vector problem (CVP), the short integer solution problem (SIS), and the learning with errors problem (LWE). SIS and LWE are closely related, and they can be built upon SVP and CVP (BOAS, 1981; AJTAI, 1996; REGEV, 2009). Typically, algorithms to solve SVP use the idea of lattice basis reduction, being often based on Lenstra–Lenstra–Lovász (LLL) (LENSTRA; JR.; LOVÁSZ, 1982), and blockwise Korkine–Zolotarev (BKZ) (SCHNORR; EUCHNER, 1994). Their approximation regimes vary, e.g., common strategies enforce enumeration, pruning, sieving, or a combination of those (SCHNORR, 2003; AONO; NGUYEN, 2017; TERUYA; KASHIWABARA; HANAOKA, 2018). Recent solutions of the Technische Universität Darmstadt Lattice Challenge show the best LWE and SVP results are being achieved by sieving-based algorithms for lattice dimensions in the 10^2 order (LINDNER et al., 2021).

Nowadays, lattice-based cryptography is said to address as many requirements as the current cryptosystems do, and more, e.g., sophisticated applications featuring homomorphic encryption (COMINETTI, 2019; ION et al., 2019). A survey on the applicability of lattice theory to cryptography, previous to the NIST standardization process, is available in (PEIKERT, 2016). Lattice-based cryptosystems hold high asymptotic efficiency, and strong provable security, backed by the computational complexity nature of lattice problems (PEIKERT, 2014). Their intrinsic computational complexity offers worst-case to average-case reduction support, meaning that, theoretically, if a given algorithm solves an average-case lattice problem, than it is possible to have that same algorithm solve a worst-case lattice problem. That may seem contradictory from an attacker perspective but it is a necessity for legitimate parts involved in a secured communication. They need attackers to see their communication data as a worst-case problem only, and, as legitimate parts have their supposedly worst-case cryptosystem modeled after an average-case problem, once these parts hold proper secrets, they manage to communicate securely, solving polynomial-time problems.

2.5.1 Schemes family: LWE

LWE cryptography is built upon hard problems from lattice theory. The family schemes derive from (AJTAI, 1996), whose asymptotic behavior is secure and reasonably efficient. Downsides for the lattice-based LWE approach include efficiency bottlenecks where DGS is present, and overly-sized keys and ciphertexts (PEIKERT; PEPIN, 2019).

A generic Ring-LWE or RLWE scheme is an LWE scheme built upon polynomial rings (LYUBASHEVSKY; PEIKERT; REGEV, 2013b; LYUBASHEVSKY; PEIKERT; REGEV, 2013a). Most RLWE cryptosystems make use of a discrete Gaussian sampling primitive to generate pseudo-random values, which by its turn can be modeled via many different sampling strategies. In the scope of this work, those values form an artificial noise vector supporting a considerable number of RLWE keys. So far, RLWE DGS imposes costly computational issues.

The CRYSTALS-Dilithium scheme or simply Dilithium is a module learning with errors (module-LWE) and module short integer solution (module-SIS) digital signature scheme, which samples from uniform distributions in order to improve its efficiency (BAI et al., 2021). By the end of the third round in the NIST post-quantum cryptography process, Dilithium was selected for standardization, being a primary recommendation for digital signature purposes (CSRC, 2019; ALAGIC et al., 2022).

The CRYSTALS-Kyber scheme or simply Kyber is a module-LWE key encapsulation or key exchange scheme, which samples from binomial and uniform distributions, in order to improve its efficiency (AVANZI et al., 2021). By the end of the third round in the NIST post-quantum cryptography process, Kyber was selected for standardization, being a primary recommendation for key-establishment purposes (CSRC, 2019; ALAGIC et al., 2022).

The NewHope scheme claims to be an RLWE-based replacement for RSA and ECC, addressing key-exchange purposes. It implements no DGS but a centered binomial distribution, in order to improve its efficiency. Authors argue DGS has a marginal impact in their scheme, being crucial only to signature and trapdoor schemes. It was a candidate for the NIST post-quantum cryptography standardization process, having been excluded by the beginning the third round of the competition (CSRC, 2019; PÖPPELMAN et al., 2019).

The qTESLA scheme is an RLWE-based digital signature scheme. It enforces DGS after a CDT strategy, for key-generation purposes only. Other than that, it uses a uniform distribution after a rejection strategy. qTESLA has been submitted to the NIST post-quantum cryptography standardization process, and as the NewHope scheme, it was excluded by the beginning of the third round of the competition (CSRC, 2019; ALKIM et al., 2019).

The FrodoKEM scheme is a plain LWE-based key encapsulation or key exchange scheme, whose DGS is modeled via CDT. Its implementation is isochronous. It lived up to the third round of the NIST post-quantum cryptography standardization process, when it failed in being selected for standardization (CSRC, 2019; ALKIM et al., 2020).

2.5.2 Schemes family: NTRU

Originally, the NTRU term referred to a single lattice-based cryptographic scheme, which was patented in 1996. The anticipated expiration for the NTRU patent occurred in 2017 (HOFFSTEIN; PIPHER; SILVERMAN, 1998; HOFFSTEIN; PIPHER; SILVERMAN, 2000; HOFFSTEIN; PIPHER; SILVERMAN, 2020). As the patent expired, cryptologic research started to benefit from it, and variant works were published. Today, NTRU-based schemes covers a variety of requirements, from key encapsulation (HÜLSING et al., 2017) to digital signatures (HOWE et al., 2019).

The Falcon scheme is based on SIS over NTRU, and it specifies a trapdoor sampling strategy. Such strategy relies on isochronous DGS routines, which are modeled via CDT and rejection sampling (FOUQUE et al., 2020). It covers digital signature requirements, making use of floating-point operations for key generation and signing. Falcon inspired ulterior schemes like MITAKA and Hawk, the latter differing from its inspiration by refraining from floating-point operations (ESPITAU et al., 2021; DUCAS et al., 2022). By the end of the third round in the NIST post-quantum cryptography process, Falcon was selected for standardization (CSRC, 2019).

2.5.3 Independent discrete Gaussian samplers

This section presents standalone samplers which are independent of schemes, as it is the case of DiGS itself.

The FACCT sampler is a general-purpose lattice-based discrete Gaussian sampler (ZHAO; STEINFELD; SAKZAD, 2020). Its sampling strategy is binary, explained in Section 2.3.3.2, and it can be integrated in both LWE-based and NTRU-based schemes.

2.6 Side-channel attacks

In a broad sense, a side-channel attack is a real-world threat to cryptographic constructions, meaning it targets cryptographic implementations. It exploits hardware and software, in order to unduly access sensitive information which otherwise would hardly be available. For example, an attack can be performed by means of probing a given CPU output, and analyzing its signal with an oscilloscope, in order to detect differences and patterns in electric signals. Risks associated to simultaneous multithreading, which is a CPU parallelization facility, constitute a notable case (LOU et al., 2021). Alternatively to the CPU, it is also possible to use exploitation vectors like the main memory, and even the network (BRUMLEY; BONEH, 2003).

2.6.1 Mitigation of timing side-channel attacks

There is a specific type of side-channel attack, known as timing, which might use signal analysis to measure times in runtime. Negligent RLWE implementations expose their samples to timing attacks. Mitigation measures to timing and other side-channel attacks can be implemented in application, operating system, and hardware levels. As an LWE-based proposal, this work addresses timing-attack risks through vulnerability-reduction measures in the application level, implementing isochrony in parts of its construction (ORTIZ, 2016; REPARAZ; BALASCH; VERBAUWHEDE, 2016).

A regular DGS implementation may be susceptible to timing side-channel attacks. Thus it is necessary to build the primitive in a way to mitigate timing risks in vulnerable routines. Achieving that is possible either through isochrony and random time insertion (ORTIZ, 2016). Isochrony in coding can be described as a design pattern which imposes an invariant execution time to a given routine. Random time insertion on the other hand adds random execution time values to vulnerable routines. DiGS, the sampler presented in this work, implements isochrony, following works that accomplish positive DGS results using invariant time routines (REPARAZ; BALASCH; VERBAUWHEDE, 2016; ORTIZ, 2016; KARMAKAR et al., 2018; ALKIM et al., 2019; HOWE et al., 2019; ALKIM et al., 2020).

2.7 Summary for general concepts

The general concepts chapter covers the algebra topics polynomial rings and lattice theory. It then presents the RLWE problem.

Next, the probability subject is covered in its distribution, sampling and CLT. The statistical metrics used are then presented.

In order to support the Fast Walsh–Hadamard Transform, a brief Discrete Fourier transform section is present.

Finally, the chapter presents recent lattice-based cryptography works, and relates them to the one presented herein. Table 2 summarizes key characteristics of the lattice-based cryptographic samplers previously presented.

Table 2: Comparison among works on lattice-based cryptography samplers.

sampler	**primitive**	**lattice assumption**	**distribution**
DiGS (ours)	digital signature	RLWE	Gaussian
Dilithium (BAI et al., 2021)	digital signature	module-LWE	uniform
FACCT (ZHAO; STEINFELD; SAKZAD, 2020)	generic	generic	Gaussian
Falcon (FOUQUE et al., 2020)	digital signature	SIS over NTRU	Gaussian
FrodoKEM (ALKIM et al., 2020)	key exchange	LWE	Gaussian
Kyber (AVANZI et al., 2021)	key exchange	module-LWE	uniform
NewHope (PÖPPELMAN et al., 2019)	key exchange	RLWE	binomial
qTESLA (ALKIM et al., 2019)	digital signature	RLWE	Gaussian and uniform

Source: author (2023).

In Table 2, grayed rows denote standalone samplers whereas regular white rows denote embedded samplers. Italicized text in column sampler indicates participation in NIST post-quantum cryptography standardization process.

3 EFFICIENT AND SECURE FWHT GAUSSIAN SAMPLER

This chapter presents the construction of an efficient and secure discrete Gaussian sampler as a primitive for digital signing in RLWE-based cryptosystems.

Efficiency gains are pursued through the CLT convergence acceleration, and also through FWHT shuffling. Security is addressed as a vulnerability mitigation to timing side-channel attacks, which is built upon algorithmic isochrony (REPARAZ; BALASCH; VERBAUWHEDE, 2016; MICCIANCIO; WALTER, 2018; HOWE et al., 2019; WANG; LING, 2019).

A generic CDT-based Gaussian sampler is also constructed for comparison purposes, following Section 2.3.3.3.

Building of approximate Gaussians is achieved by means of sampling algebraic Gaussian distributions, instead of uniform distributions, meaning a biased procedure is enforced. This is also the phase of CLT convergence, which consists in getting to a probability distribution fitting in reduced time. The number of sampling iterations are reduced by a precision acceptability criteria, which estimates tolerance values for differences between the mean of the polynomial roots and the roots of the algebraic polynomial (BHARUCHA-REID; SAMBANDHAM, 1986), offering opportunities for routines to be terminated prematurely, still being effective. The FWHT is then used for shuffling, as shown in 2.4.1.3. Sampling is always performed in a fixed regime, i.e., Gaussian parameter values do not change during a sampling session.

Metrics regarding efficiency and approximation quality are used. The term first-level comparison denotes a comparison between a PDF and its resulting PMF, for a given sampling strategy, and the term second-level comparison denotes a comparison between results of distinct sampling strategies.

Regarding the structure of this chapter, the first section introduces the many variables dealt with herein. Then, section 3.2 details the FWHT-based random number generator, Section 3.2.1 presents mathematical formalization, and Section 3.3 presents implementation details for computer routines. In 2.4.1.3, pseudocode and diagrams, related to the implemented routines, are presented.

3.1 Variables

Variables may have an $\mathcal{N}$ index if they relate to the algebraic normal distribution PDF, or a $\mathcal{D}$ index if they relate to the approximate Gaussian PMF. Additionally, the following presentation of variables divides them in two groups, input and output, all of them are presented in the following sections. Values of input variables are supposed to be secret.

Input variable: $\sigma_{\mathcal{N}}{}^{2}$ denotes the variance for a Gaussian PDF. Its square root is the standard deviation σ, which is used to determine where to bound the PDF, for practical purposes. Standard deviation is directly proportional to the Gaussian parameter S, as presented by Equation 3.1:

$$\sigma = \frac{S}{\sqrt{2\pi}} \tag{3.1}$$

Many works present the normal distribution standard deviation or variance as relevant parameters for discrete Gaussian sampling (DWARAKANATH; GALBRAITH, 2014; FOLLÁTH, 2014).

Input variable: $\tau_{\mathcal{N}}$ is the tail-cut factor. It is not often used for simple Gaussian descriptions, but it constitutes an artificial resource for bound manipulation.

Input variable: β is a scalar, $\beta \in \mathbb{N}^*$, created to comply with dimension requirements of Hadamard matrices. The variable influences all of the sampling routines, both on PDF and PMF functions. This is because the FWHT-based sampling strategy demands a 2^β number of positions on its input vector. See Section 2.4.1.3 for further details on the transform. As a consequence, not only does the β exponent dictate the number of stages in the FWHT algorithm presented by this work, but it also determines a 2^β order square matrix, the full-rank lattice dimension to work with, and the number of keys to be supported. In practical terms, β influences the sample size, that is, the number of observations in each sample, and it also influences the number of samples itself. Thus, a β value of 8 produces a sample size and a number of samples of 256. Regarding the sample size, denoted in this work as s, each observation corresponds to one polynomial coefficient, that is, a sample size of 256 implies a polynomial of degree 255 with 256 coefficients. As for the number of samples, each sample represents a polynomial, so 256 samples stand for 256 polynomials.

Output variable: $\sigma_{\mathfrak{D}}$ denotes the standard deviation for the Gaussian PMF. The formula used is presented in Equation 3.2.

$$\sigma_{\mathfrak{D}} = \sqrt{\frac{1}{2^\beta}\sum_{i=1}^{2^\beta}(D_i - \mu_{\mathfrak{D}})^2} \tag{3.2}$$

Output variable: $\mu_{\mathfrak{D}}$ denotes the mean of a normal distribution PMF.

Equation 3.3 calculates the mean for 2^β samples.

$$\mu_{\mathfrak{D},2^\beta} = \frac{\sum_{i=1}^{2^\beta} D_i}{2^\beta} \tag{3.3}$$

Output variable: s lowercase s is the even 2^β sample size to be used both by PDF and PMF sampling routines, and it also represents a security parameter of the RLWE problem (LYUBASHEVSKY; PEIKERT; REGEV, 2013b). The sample size is the number of observations in each sample, and, in this work, that also means the number of coefficients in each RLWE polynomial. Bigger values are expected to reduce tails, and make statistical distance smaller between algebraic PDF and approximated PMF (DWARAKANATH; GALBRAITH, 2014).

Output variable: $skewness_{\mathfrak{D}}$ denotes quality. An approximated normal distribution PMF whose skewness value is closer to zero indicate a more likely bell-shaped function. The already adapted formula used here is presented in Equation 3.4.

$$skewness_{\mathfrak{D}} = \frac{\sum_{i=1}^{2^\beta}(D_i - \mu_{\mathfrak{D}})^3\, PMF[(D_i - \mu_{\mathfrak{D}})^3]}{{\sigma_{\mathfrak{D}}}^3} \tag{3.4}$$

Output variable: $kurtosis_{\mathfrak{D}}$ as $skewness_{\mathfrak{D}}$, the $kurtosis_{\mathfrak{D}}$ variable stands as a measure of quality. Higher values of kurtosis indicate lower-quality PMF approximation. The formula used is presented by Equation 3.5.

$$kurtosis_{\mathfrak{D}} = \frac{\sum_{i=1}^{2^\beta}(D_i - \mu_{\mathfrak{D}})^4\, PMF[(D_i - \mu_{\mathfrak{D}})^4]}{{\sigma_{\mathfrak{D}}}^4} \tag{3.5}$$

Output variable: $\boldsymbol{CPUcycles_{\mathcal{N}}}$ portrays an efficiency metric, reporting the number of CPU cycles spent until CLT convergence. Typically, it can be calculated by a formula like that of Equation 3.6.

$$CPUcycles = frequency \cdot time \tag{3.6}$$

In Equation 3.6, $frequency$ represents a CPU clock frequency, given in hertz (Hz), and $time$ represents the time spent in a computer routine, given in seconds (s).

Output variable: $\boldsymbol{CPUcycles_{\mathfrak{D}}}$ portrays an efficiency metric, reporting the number of CPU cycles spent in the production of a noise vector. As it is the case for $CPUcycles_{\mathcal{N}}$, $CPUcycles_{\mathfrak{D}}$ can be calculated by a formula like that of Equation 3.6.

Output variable: $\boldsymbol{CPUcycles_{total}}$ portrays an efficiency metric, corresponding to the total CPU cycles, from the moment routines start trying to achieve CLT convergence until the production of a noise vector, that is

$$CPU\,cycles_{total} = CPU\,cycles_{\mathcal{N}} + CPU\,cycles_{\mathfrak{D}}. \tag{3.7}$$

Output variable: $\boldsymbol{entropy}$ is also known as the Shannon entropy. For the purposes of this work, higher values of this variable are desirable, as they offer more randomness. Given in bits, the maximum entropy value possible for a 2^{β} sample size is described by Equation 3.8.

$$entropy \leq \log_2 2^{\beta} \tag{3.8}$$

Equation 3.8 shows that, for the purposes of this work, the maximum entropy possible is less than or equal to β bits.

Output variable: D_{KL} is the relative entropy, measured in bits. Values closer to zero are desirable in this work, as they stand for PMF functions better representing the original PDF function. Equation 3.9 describes how the relative entropy is computed from a continuous-case distribution to a discrete-case one.

$$D_{KL}(\mathcal{D}||\mathcal{N}) = \sum_{i=1}^{2^{\beta}} \log(\frac{\mathcal{D}_i}{\mathcal{N}_i}) \tag{3.9}$$

3.2 Sampler modeling

Theoretical modeling background for the FWHT-based DGS sampler is presented in this section.

The FWHT routine presented in Algorithm 3.1 is a contribution of this work. It is supposed to be provided with argumentative values for the following parameters: the exponent β, the $\sigma_{\mathcal{N}}$ standard deviation for the Gaussian PDF, the $\tau_{\mathcal{N}}$ tail-cut factor, and the $option$ sampling algorithm. As for the algorithm variables, ℓ is expected to store the number of cycles on a given stage. Variable ℓl is expected to store the current cycle length, that is, the number of vector positions the current cycle takes. Variable ℓlh is expected to store half the cycle length, and it also helps didactically in the pseudocode, emphasizing the idea cycles always have two equal-sized chunks, one for addition operations and other for subtraction operations. Variable v is a 2^{β}-

sized vector, which firstly stores an η initialization vector, produced by the $sample()$ function, and the PDF sampling strategy to be performed by this function is arbitrated by the $option$ parameter. Variable v_{aux} serves the purpose of preserving the value stored in vector position $\ell_{aux}\ell l + \ell l_{aux}$ before it is altered. Variables β_{aux}, ℓ_{aux}, and ℓl_{aux} help in counting and controlling the number of iterations in each of the three loop structures. Variable ℓp is expected to store a relative position in vector v, always in the first chunk of the current cycle, computed as $\ell_{aux}\ell l + \ell l_{aux}$.

Algorithm 3.1 An FWHT isochronous construction.

```
algorithm fwht_non_recursive ($\beta$: natural, $\sigma_N$: real, $\tau_N$: real, $option$: natural)
    $\ell$:                          natural; // Number of cycles on a given stage.
    $\ell l$:                         natural; // Current cycle length.
    $\ell lh$:                        natural; // Half the cycle length.
    $\ell p$:                         natural; // Position in the first chunk of the current cycle.
    $v[2^\beta]$, $v_{aux}$:             integer;
    $\beta_{aux}$, $\ell_{aux}$, $\ell l_{aux}$:     natural;

    $v$                          $\leftarrow sample(\beta, \sigma_N, \tau_N, option)$;
    $\ell$                          $\leftarrow 1$;
    $\ell l$                         $\leftarrow 2^\beta$;

    for ($\beta_{aux}$ from $0$ to $\beta - 1$, step $1$)
        $\ell$                      $\leftarrow \ell \cdot 2^{\beta_{aux}}$;
        $\ell l$                     $\leftarrow \ell l / 2^{\beta_{aux}}$;
        $\ell lh$                    $\leftarrow \ell l / 2$;
        for ($\ell_{aux}$ from $0$ to $\ell$, step $1$)
            for ($\ell l_{aux}$ from $0$ to $\ell lh - 1$, step $1$)
                $\ell p$              $\leftarrow \ell_{aux} \ell l + \ell l_{aux}$;
                $v_{aux}$             $\leftarrow v[\ell p]$;
                $v[\ell p]$           $\leftarrow v_{aux} + v[\ell p + \ell lh]$;
                $v[\ell p + \ell lh]$ $\leftarrow v_{aux} - v[\ell p + \ell lh]$;
            end for
        end for
    end for

    return $v$;
end algorithm
```

Source: author (2022).

Algorithm 3.1 presents a non-recursive isochronous FWHT construction, using imperfectly-nested loop nests (KODUKULA; PINGALI, 1996), with the purpose of mitigating timing vulnerabilities. Vector v has its content successively overwritten throughout the iterated β stages, and it is also returned with the $g(\omega)$ final stage calculation result.

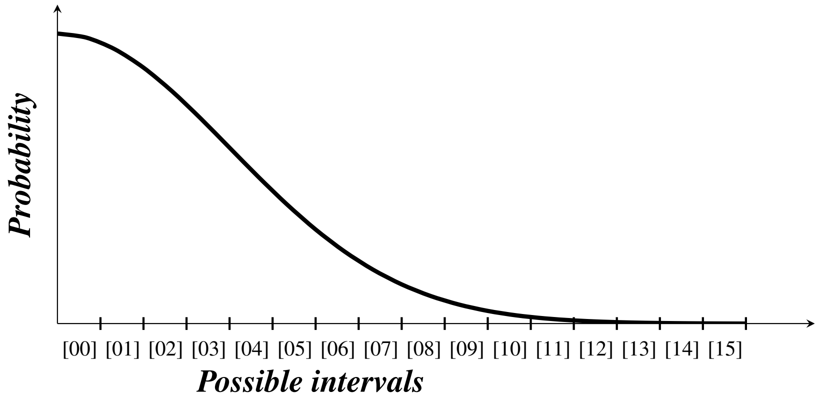
The first loop structure iterates β times, corresponding to the number of stages. The second loop structure iterates on the number of cycles of the current stage. In the first stage, there is only one cycle. Then, in the second stage, there are two cycles, and that value keeps being doubled at each new stage. The third and last loop is based on the length of the current cycle, but it iterates upon only half that value. In the first stage, the cycle length equals 2^{β}. In the second stage, the cycle length becomes $2^{\beta-1}$. Then $2^{\beta-2}$ in the third stage, and the exponent keeps being decremented by one at each new stage, until it reaches the value of one, meaning the last stage has cycles with 2 elements. Thus, as the number of cycles double, the number of elements in each cycle decreases in an exponential rate. Using only half the cycle length for iteration is due to the FWHT always breaking a cycle length into two equal-sized chunks. In this work, a chunk corresponds to half the length of a cycle. For each chunk element, there is a sign attribution, followed by an addition or a subtraction operation, respectively represented by lines 21 and 22 of Algorithm 3.1. Line 21 represents the first of the two chunks, so its first position corresponds to the first position of the current cycle, which is determined by ℓp. In the first chunk, vector elements are given plus signs, and then each of those elements is used in two operations, the first operation occurring in the first chunk itself, and the second operation occurring in the second chunk. Line 22 represents the second of the two chunks, so its first position corresponds to the middle of the current cycle, which is determined by $\ell p + \ell lh$. In the second chunk, vector elements are given plus signs for operations in the first chunk, and minus signs for operations in the second chunk itself.

Lastly, v is returned as the $g(\omega)$ transform output. It has the same 2^{β} length of the η input, and its content is a set of integer coefficients for a polynomial. Those integer values may be turned into binaries if necessary. Possible options to generate the η

input, and to convert the FFT output, are discussed in Chapter 5.

In the example represented by Figure 9 and Figure 10, didacticism is addressed, and many one-valued positions are used in the η vector. However, for the sampling routine of this work, there is a single non-zero value in the referred IV, all other positions being occupied by zeros. The η initialization vector is obtained from a preliminary Gaussian sampling observation. Thus, still considering the $\beta = 4$ example, the referred sampling is performed in the positive portion of a $\mathcal{N}(0, \sigma^2)$ truncated PDF, whose domain is divided into 16 equally-sized intervals, as that of Figure 11. Each interval corresponds to one η vector position.

Figure 11: The possible intervals to be observed for $\beta = 4$.

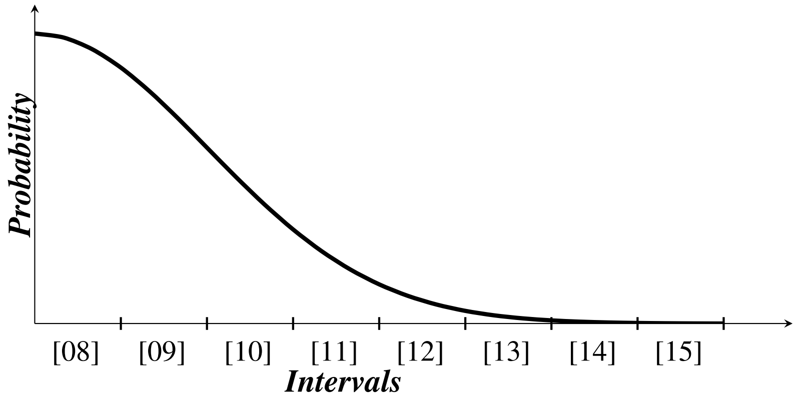


Source: author (2021).

In this work, the μ mean assumes a value of zero, and the positive bound, $\tau\sigma$. Thus, a sampling like that of Figure 11 uses a $(0, \tau\sigma]$ domain, and observed values assume alternating signs minus and plus. Thus, in practice, half the number of intervals is used, as in Figure 12, and the alternating signs simulate a $(-\tau\sigma, \tau\sigma]$ domain. As an exception is forced in the negative bound, the first sign is chosen to be minus.

In Figure 12, intervals corresponding to positions [00], [01], [02], [03], [04], [05], [06], and [07] are not shown because they stand in the negative side of the curve. A negative [08] corresponds to [07], a negative [09] corresponds to [06], a negative [10] corresponds to [05], a negative [11] corresponds to [04], a negative [12] corresponds to [03], a negative [13] corresponds to [02], a negative [14] corresponds to [01], and a

Figure 12: The intervals to be observed for $\beta = 4$.



Source: author (2021).

negative [15] corresponds to [00].

Previously to observations, the vector is filled with zeros. Then, after having one of its positions occupied by 1, such position corresponding to the observed interval, η is subjected to the FWHT, in order to generate a $g(\omega)$ transform representation.

For the purpose of obtaining initial standard deviation values to start modeling DGS, (ALBRECHT et al., 2018; ALBRECHT et al., 2019) are used. Results, presented in Table 3, are picked for available schemes in the interest of this work, which are Falcon, FrodoKEM, and qTESLA. They all make use of isochronous and discrete Gaussian sampling. The primal attack is used as a filter because its results are available for both LWE and NTRU lattice assumptions, the referred attack being a solver variant for Shortest Vector Problem (SVP) lattice problems. Column security represents the security level, β is the exponent for the number of samples, and σ is the standard deviation.

Table 3: Comparison of NIST lattice-based candidates.

scheme	**security (bits)**	β	σ
Falcon	103	512	4.05
FrodoKEM	150	976	2.30
qTESLA	128	1024	8.49

Source: author (2021).

Although the Falcon signing scheme is built upon NTRU lattice assumptions, the works (ALBRECHT et al., 2018; ALBRECHT et al., 2019) test it with LWE premises. Thus, it should be pointed out that the first row in Table 3 refers to numbers from those LWE-adapted Falcon tests.

Variance of the PDF: initial $\sigma_{\mathcal{N}}$ value is 4.9467, obtained by computing the mean for the three σ values present in Table 3. Thus, initial variance is $4.9467\ ^2$, that is, 24.4695.

Mean of the PDF: PDF functions in this work are constantly centered in zero, meaning their mean value, $\mu_{\mathcal{N}}$, is fixed to zero.

Bounds of the PDF: as explained in Section 2.3.1.1, the PDF functions of this work are truncated in the interval $[-\tau_{\mathcal{N}}\sigma_{\mathcal{N}},\ \tau_{\mathcal{N}}\sigma_{\mathcal{N}}]$. Regarding the PDF tail-cut factor $\tau_{\mathcal{N}}$, in (KARMAKAR et al., 2018), authors propose values of $\tau_{\mathcal{N}}$ between six and twelve, and in (ZHAO; STEINFELD; SAKZAD, 2020), tail-cut factor values are said to be between ten and twelve. Here, initial value for the input variable $\tau_{\mathcal{N}}$ is ten.

Security level: the discrete Gaussian sampler presented by this work enforces a 128-bit security level, meaning PMF functions should offer 2^{128} possible values in their considered intervals. PMF bounds and output variable $\sigma_{\mathfrak{D}}$ should produce values to comply with such level. Considering the integers necessary to offer 128-bit resolution, if PMF is centered in zero, a $(-2^{127}, 2^{127}]$ domain interval supports such resolution. Also, the referred security level implies in a 128-bit valued seed.

Number of samples, and sample size: the sampler assumes a full-rank lattice, which means the number of samples is equal to the number of observations inside each sample. A square matrix is to be built, and its order should meet Hadamard matrices requirements, meaning both the number of samples, and the sample size, are computed as 2^{β}, and β initial value is one. That same value determines the size of the e error vector, and ultimately, the number of RLWE keys covered. Preliminary tests indicate feasibility of computation for β values up to twenty.

3.2.1 Mathematical formalization

This section uses the RLWE problem presented in 2.2. For the efficiency purpose of this work, only one random variable needs to be formed by sampling. Others can be derived from the first (RADER, 1969). This strategy could be applied to key generation but in this case, we are generating noise.

Initially, concerning the statistical model, it consists in a bounded normal probability distribution with zero mean, which is enforced as the sampling source. The relation between lattice-based cryptosystems and sampling from truncated normal distribution PDF functions is close enough to make works like the estimator of (ALBRECHT et al., 2018) assume every error vector distribution to be a discrete normal, even if they are normal or binomial distributions. Since the FWHT is used, there is a requirement for a 2^{β} sample vector size, $\beta \in \mathbb{N}$. As for context, we consider such vector as an element of a 2^{β} -dimension lattice, thus we also need 2^{β} samples or polynomials, constituting a polynomial ring. Also, a 2^{β} lattice dimension means polynomials having a degree of $2^{\beta} - 1$.

The data structure into which samples are stored is a square matrix A of order 2^{β}, and its inverse is later denoted as A^{-1}.

From the e error vector in Equation 2.8, which is supposed to harden the system solution, it is possible to write the matrix inequation in 3.10:

$$\begin{bmatrix} x^{2^\beta-1} \\ x^{2^\beta-2} \\ \vdots \\ x^2 \\ x \\ 1 \end{bmatrix} + \begin{bmatrix} e_{1\,1} \\ e_{2\,1} \\ \vdots \\ e_{2^\beta-2\,1} \\ e_{2^\beta-1\,1} \\ e_{2^\beta\,1} \end{bmatrix} \neq \begin{bmatrix} c_{1\,1} & c_{1\,2} & \cdots & c_{1\,2^\beta-2} & c_{1\,2^\beta-1} & c_{1\,2^\beta} \\ c_{2\,1} & c_{2\,2} & \cdots & c_{2\,2^\beta-2} & c_{2\,2^\beta-1} & c_{2\,2^\beta} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ c_{2^\beta-2\,1} & c_{2^\beta-2\,2} & \cdots & c_{2^\beta-2\,2^\beta-2} & c_{2^\beta-2\,2^\beta-1} & c_{2^\beta-2\,2^\beta} \\ c_{2^\beta-1\,1} & c_{2^\beta-1\,2} & \cdots & c_{2^\beta-1\,2^\beta-2} & c_{2^\beta-1\,2^\beta-1} & c_{2^\beta-1\,2^\beta} \\ c_{2^\beta\,1} & c_{2^\beta\,2} & \cdots & c_{2^\beta\,2^\beta-2} & c_{2^\beta\,2^\beta-1} & c_{2^\beta\,2^\beta} \end{bmatrix} \times \begin{bmatrix} -a_{1\,2^\beta} \\ -a_{2\,2^\beta} \\ \vdots \\ -a_{2^\beta-2\,2^\beta} \\ -a_{2^\beta-1\,2^\beta} \\ -a_{2^\beta\,2^\beta} \end{bmatrix}. \tag{3.10}$$

The discrete Gaussian sampler presented herein generates the e vector from a truncated normal distribution.

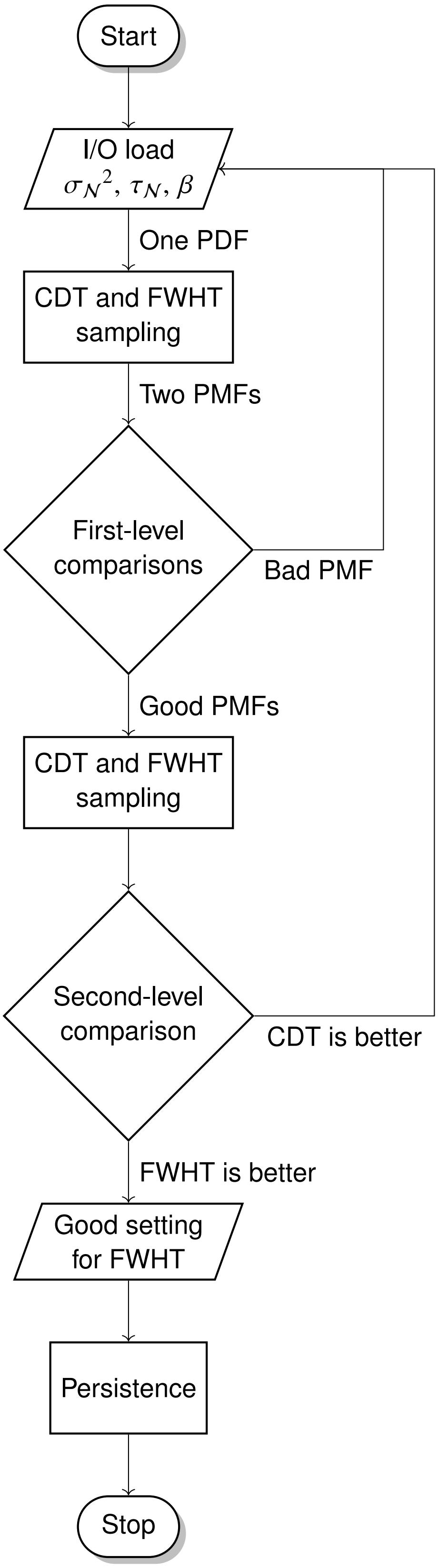
For the main tests, values of β varies from 0 to 23, meaning that in the context of this work, the maximum number of polynomials to deal with in a system like 2.4, is 2^{23}.

3.3 Specification and implementation of computer routines

This section presents implementation information for DiGS, which is an interactive computer routine, having evolved from a SageMath script to a Python 3 script, and also to a C language program, which is depicted by the flowchart in Figure 13. The C programming language standard used is C99. Even though the standard has been replaced, it is almost fully supported by the GNU Compiler Collection (GCC), which in turn is the software chosen for compilation (Technical Committee: ISO/IEC JTC 1/SC 22, 1999; GCC Team, 2023).

Its inputs are algebraic Gaussian probability distributions, and outputs are intended to be approximate Gaussian probability distributions. The referred script was initially ported to the Python language, and the concept was verified as consistent. Then it was

Figure 13: DiGS routine flowchart.



Source: author (2023).

rewritten in the C language, with improvements. The program builds discrete Gaussian sampling primitives following two distinct strategies, namely CDT, and FWHT shuffling.

Input variable values, e.g., those listed under Section 3.1, may be provided in runtime. However, the code offers default values, for quick exemplifying.

Efficiency remarks: regarding the FWHT, improvements are built upon avoidance of floating-point computations, restricting operations to sums and subtractions only.

Isochrony remarks: isochrony is implemented with an invariant number of steps during shuffling iterations in computing routines.

Sampling remarks: the coefficient matrix is implemented as a simple C array data structure with pointers because it is generally considered faster for regular computers (SUMMIT, 2019). FWHT construction in DiGS receives a 2^{β}-sized array of pointers, pointing to binary values, as input. Variable β is described in 3.1. Assuming initial parameter values set, the routine is described next, respecting the steps in Section 1.5. Samples are generated and then processed, as described in Section 1.5.1. Firstly, histograms are built, and then, interpolated Gaussian-like PMF functions too. A simple example of that process is present in A.2. For quality assurance of CDT and FWHT strategies, the PDF used in the last step is compared to each of the produced PMF functions, using the statistical metrics presented in Section 3.1. The first comparison round is referred to in this work as a first-level comparison. It embraces a comparison between the original PDF and the computed CDT PMF, and a comparison between the original PDF and the computed FWHT PMF. Results considered as unacceptable force the whole dataset to be discarded, and the process start over. In that case, one or more of the initial values defined in the prototyping step of Section 1.5.1 may be changed, until acceptable statistical values are obtained. Once acceptable statistical results are achieved, the two distinct PMF functions are available for sampling. Then, CDT sampling is performed on the CDT PMF, and FWHT sampling is performed on the FWHT PMF. Results are compared, and that is referred to in this work as the second-level comparison. If no FWHT efficiency advantage nuance is present, former steps are reviewed, and variations might be tested, e.g., different initial values. Finally, the data persistence takes place.

DiGS FWHT routine outputs another array of pointers, with the same length of the one received as input, but this time pointing to integers. DiGS grabs the FWHT output, and in order to produce its binary noise vector, it turns even numbers in zeroes, and odd numbers in ones. Then, FWHT input and output are then destroyed for security.

Yet, there is an additional remark regarding $g(\omega)$, for the conversion of its contents to zeros and ones. According to (MARINGER; FRITZMANN; SEPÚLVEDA, 2019), β influences stochastic dependence of LWE cryptosystems in a directly proportional manner, that is, for the purposes of this work, bigger values of β would tend to reduce randomness. Indeed, it is observed that bigger β values in the FWHT DGS make parity predictability grow on the output vector. Its numbers tend to be even. That does not impose a problem to DiGS because its binary vector of noise is built upon k, $k \in \mathbb{Z}$, which by its turn form all even numbers, as $2k$.

Execution remarks: in order to run it on a Linux operating system command line interface, considering one is already on the directory containing the executable file, then in a regular bash shell, DiGS should be manually invoked as:

```
$ ./digs
```

3.3.1 Calculation precision

In regular computer routines, calculation precision is limited by the largest number registers can hold. That is sometimes called fixed-size arithmetic. DiGS routines deal with numbers bigger than those supported by regular registers. So it would be affected by the aforementioned limitation. Two options of library categories were considered to circumvent such issue:

i) **arbitrary-precision arithmetic**

   sometimes called multiple-precision arithmetic, as opposed to fixed-size arithmetic, arbitrary-precision arithmetic libraries work with floating-point numbers; and

ii) **symbolic computation**

   this category of library do not work with floating-point numbers, but symbols.

In the Python proof-of-concept (PoC) phase, the following libraries were considered:

- **decimal**

  Python-native library;

- **gmpy**

  Python version of the GNU Multiple Precision Arithmetic Library (GMP) program; and

- **mpmath**

  Python-native library, which was the choice during the Python development phase, for providing an easy interface to work with gmpy. The mpmath library can be found alone or bundled in the larger SymPy library (JOHANSSON et al., 2013).

Afterwards, in the C development phase, the following arbitrary-precision arithmetic libraries were considered:

- **GNU Multiple Precision Arithmetic Library (GMP)**

  the choice for this work is a portable library written in C, and it works with integers, rational numbers, and floating-point numbers;

- **libgcrypt**

  a cryptology library which has its arbitrary-precision arithmetic routines as a fork of an old GMP release; as a part of the GnuPG project, it is adapted to suit its main project requirements; and

- **RELIC**

  a library for working with integers (ARANHA; GOUVÊA, 2020).

3.4 Summary for efficient FWHT Gaussian sampler

Development of an efficient Gaussian sampler, and a method to measure it, are present in the current chapter. It details how to improve DGS, and the full set of variables used to achieve it. Mathematical background is also provided, as well as the computer routines specifications.

4 TESTS AND RESULTS

Results for the discrete Gaussian sampling tests with computer routine software implementations are presented in this chapter. The tests are executed with the cumulative distribution table strategy, and with the fast Walsh–Hadamard transform strategy. Initial setting for both samplers is described in Section 3.2, and the CDT itself is presented in A.3. The referred routines compute the metrics specified in Section 2.3.5. Testing sequence is detailed as enumerated steps in Section 1.5. The continuous-case algebraic normal probability distribution used is $\mathcal{N}(0, 14.71025358)$.

Table 4 presents the results for the CDT sampling strategy . For all of the listed β values, skewness is zero and kurtosis is -2. Relative entropy values are truncated for cleaner presentation.

Table 4: Results for the cumulative distribution table strategy.

β	s	**CPU cycles ×10^6**	$\mu_{\mathcal{D}}$	$\sigma_{\mathcal{D}}$	**relative entropy**
1	2	16.40	21.0	15.0	∞
2	4	16.42	-1.5	4.5	∞
3	8	16.40	9.0	19.0	∞
4	16	14.35	7.0	12.0	0.346
5	32	14.35	4.0	17.0	0.415
6	64	13.32	11.0	10.0	0.485
7	128	12.81	-9.0	17.0	0.346
8	256	12.82	-8.0	10.0	0.462
9	512	13.95	-1.0	12.0	0.356
10	1024	13.72	-13.5	2.5	0.373
11	2048	13.49	-13.0	8.0	0.360
12	4096	13.41	-4.5	10.5	0.353
13	8192	13.60	-1.0	10.0	0.366
14	16384	13.31	-22.0	1.0	0.360
15	32768	13.33	4.0	26.0	0.355
16	65536	14.35	-2.0	2.0	0.359
17	131072	14.13	-3.5	23.5	0.363
18	262144	14.34	-12.5	7.5	0.359
19	524288	14.20	3.5	2.5	0.359
20	1048576	14.19	-17.0	15.0	0.360

Source: author (2021).

Table 5 presents the results for the FWHT sampling strategy . For all of the listed β values, the obtained μ, is zero. The same applies to skewness values. Relative entropy values are truncated for cleaner presentation.

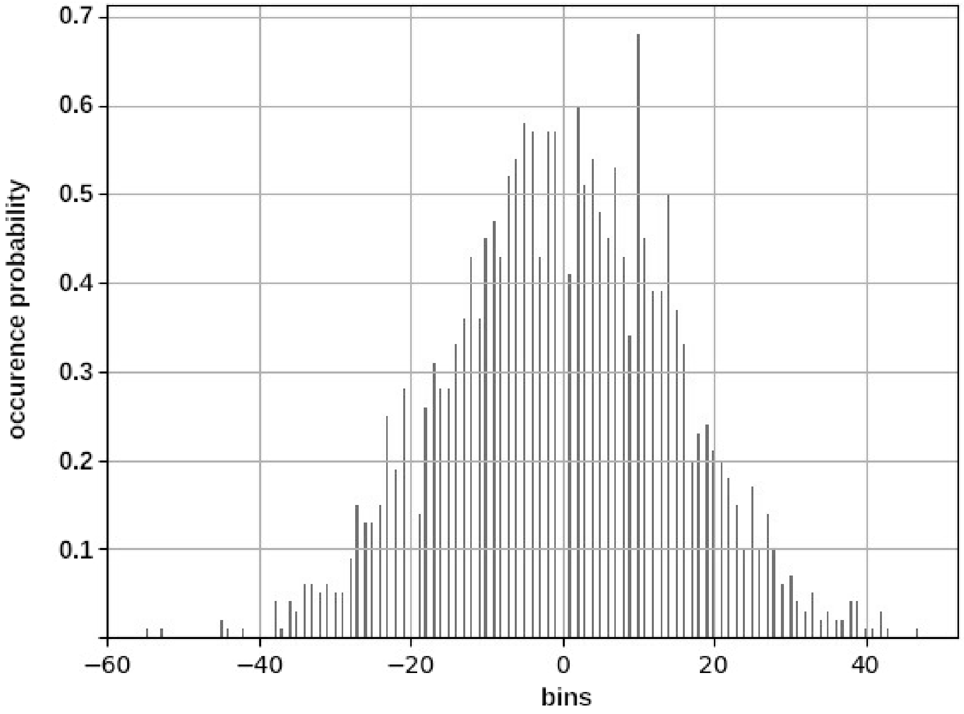
Table 5: Results for the fast Walsh–Hadamard transform strategy.

β	s	**CPU cycles ×10^3**	$\sigma_{\mathcal{D}}$	**kurtosis**	**relative entropy**
4	16	0.00	18	-2	0.277
5	32	0.00	17	-2	0.207
6	64	0.00	11	-2	0.306
7	128	0.00	10	-2	0.346
8	256	128.10	2	-2	0.346
9	512	64.05	12	-2	0.306
10	1024	96.18	2	-2	0.301
11	2048	96.18	2	-2	0.333
12	4096	104.16	14	-2	0.332
13	8192	110.67	12	-2	0.325
14	16384	124.11	10	-2	0.316
15	32768	114.24	10	-2	0.320
16	65536	111.72	6	-2	0.319
17	131072	111.51	15	-2	0.320
18	262144	109.83	8	-2	0.317
19	524288	111.72	0	-3	0.318
20	1048576	109.20	18	-2	0.317
21	2097152	108.57	7	-2	0.317
22	4194304	107.73	2	-2	0.318
23	8388608	108.36	12	-2	0.318

Source: author (2021).

A histogram plot corresponding to FWHT tests is present in Figure 14.

Figure 14: Histogram of observations for the fast Walsh–Hadamard transform sampling strategy.



Source: author (2021).

4.1 Summary for tests and results

In this chapter, quantitative content is presented in an organized manner. Table 6 presents some of the results, combining both strategies side by side, for comparison convenience. In this comparison, the β discrete domain used starts in 4 because lower values produce unwanted relative entropy. Then it ends in $\beta = 20$ because bigger values proved to be computationally costly regarding the CDT strategy, given the context of this work. Values are subjected to approximations, in order to make a cleaner presentation.

Discussion about Table 6 is present in Chapter 5, and complete tables are present

Table 6: Comparison of results.

	$\sigma_{\mathfrak{D}}$		$CPU\ cycles_{N}$		**relative entropy**	
β	**CDT**	**FWHT**	**CDT ×10^6**	**FWHT ×10^3**	**CDT**	**FWHT**
4	12.00	18.00	14.35	0.00	0.34	0.28
5	17.00	17.00	14.35	0.00	0.41	0.21
6	10.00	11.00	13.32	0.00	0.48	0.30
7	17.00	10.00	12.81	0.00	0.34	0.34
8	10.00	2.00	12.82	128.10	0.46	0.34
9	12.00	12.00	13.95	64.05	0.35	0.30
10	2.50	2.00	13.72	96.18	0.37	0.30
11	8.00	2.00	13.49	96.18	0.36	0.33
12	10.50	14.00	13.41	104.16	0.35	0.33
13	10.00	12.00	13.60	110.67	0.36	0.32
14	1.00	10.00	13.31	124.11	0.36	0.31
15	26.00	10.00	13.33	114.24	0.35	0.32
16	2.00	6.00	14.35	111.72	0.36	0.32
17	23.50	15.00	14.13	111.51	0.36	0.32
18	7.50	8.00	14.34	109.83	0.36	0.32
19	2.50	0.00	14.20	111.72	0.36	0.32
20	15.00	18.00	14.19	109.20	0.36	0.32

Source: author (2021).

in Appendix A.

5 CONCLUSIONS

Like many other fields, cryptology is being pushed forward by quantum computing, and the efforts spent in this work represent a contribution to further advancements. Schedule delays were caused by the COVID-19 pandemic consequences but the goal, as described in 1.3, is successfully achieved. Preliminary results were presented in VIII Workshop de Pós-Graduação em Engenharia de Computação, and published in the corresponding proceedings, in (JÚNIOR; JUNIOR, 2019). Final results were presented in XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais, and published in the corresponding proceedings, in (JÚNIOR, 2022). As the NIST process is still in course, fresh material related to lattice-based cryptography is often published. Efforts are made in order to work with accurate and updated information as much as possible.

As for the input parameters, the β exponent determines the most relevant aspects of the sampler, from lattice dimension, and polynomial degrees, to CLT convergence rate, the entropy of samples, and the key size to be supported. Additionally, in terms of implementation, β is directly responsible for the CPU cycles spent. Modeling DGS via the β parameter alone reveals itself as a challenge in searching for optimal values, because higher values raise entropy, but they also lower efficiency. Thus, the β exponent represents a trade-off involving efficiency and security. Natural values ranging from zero (0) to twenty (20) are tested for both CDT and FWHT. Still, the values of twenty one (21), twenty two (22) and twenty three (23) are tested for the FWHT case only. Given the context and setup of this work, those β exponent values prove to be unfeasible for CDT.

The standard deviation $\sigma_{\mathcal{N}}$ of the PDF is an input value, which influences inflection points of the function. In cases of bounded domains, it is possible to affirm that $\sigma_{\mathcal{N}}$ also influences the cardinality of the sample space. For the settings of this work, the optimal standard deviation value of the PDF is $\sigma_{\mathcal{N}} = 14.71025358$.

The tail-cut factor $\tau_{\mathcal{N}}$ of the PDF was initially planned to be used as an input parameter because, along with $\sigma_{\mathcal{N}}$, it influences the cardinality of sample spaces. However, acceptable efficiency results are obtained without it, thus, for the sake of simplicity, this work chooses not to use it.

As for the outputs, feasible lattice dimensions of up to 2^{23}, that is, $8,388,608$, are covered herein. Regarding $\sigma_{\mathcal{D}}$ values, Table 6 shows that in most cases, the obtained numbers are more than 10% above or below $\sigma_{\mathcal{N}}$. That is not considered as a good approximation result for this work.

Regarding the Kullback–Leibler divergence values, the fact that all measures are closer to zero than they are to one denotes a reasonable approximation achievement. Such values indicate PMFs resemble their respective PDFs.

Regarding the quality of the PMFs, there are the results for coefficient of skewness, and kurtosis. All of the obtained coefficient of skewness values are zero, showing maximum quality is achieved for that metric. Kurtosis is always -2 in all tests but one, in which a value of -3 is obtained, corresponding to the FWHT test for $\beta = 19$. Approaching zero from the left, with absolute values below 3 means the PMFs are platykurtic, i.e., they are flatter than they should be.

As for the lattice-based approach, some conceptual coherence is perceived in its cryptologic utility. This is due to the periodic structure of lattices adhering to generic modulus specifications often used in cryptology. Regarding lattice dimensions, the highest values solved, by the time of this work, are in the 10^2 order, but specifically for LWE, the biggest dimensions solved do not reach the value of one hundred (LINDNER et al., 2021). In this work, which is based on the LWE problem, sampling works with lattice dimensions in the 10^6 order.

As for practical applications, this work specifies and implements a discrete Gaussian sampler for lattice-based cryptosystems. Also, with a few adaptations, it can generate full sets of polynomial coefficients if needed, covering sets of up to 2^{23} RLWE keys. The sampler intends to be useful for RLWE-based digital signature schemes. DiGS implementation is planned to be available as a C language library, e.g., the generated e error vector, presented in Equation 3.10, may be easily returned to cryptosystems used in the context described.

5.1 Efficiency conclusions

DGS is considered expensive both in computational and memory terms, if compared to, e.g., uniform sampling, which causes some works to reject the former or to use it moderately, as seen in section 2.5. That is due to DGS extensive low-level evaluation of integrals or series through specialist algorithms. As a typical FFT algorithm, the fast Walsh–Hadamard transform (FWHT) is $O(n \cdot \log n)$. Such complexity holds for the isochronous shuffling algorithm of this work, which is $O(\beta \cdot 2^{\beta})$, if expressed through the β exponent.

As for alternative constructions, a feasible workaround to avoid isochronous constructions and keep implementations secure is to insert random delay times in timing-vulnerable routines. That alternative may be worthwhile if the random delay time generation alternative takes less time than using the invariant routine. Greater delay values could be detected, which would jeopardize the referred workaround. Also, as it can be seen in the diagram of Figure 9, parallelization may be an option, e.g., from the second stage on, cycles belonging to the same stage do not depend on each other. Notwithstanding, simultaneous multithreading may also increase susceptibility to side-channel attacks.

As for the comparison between FWHT and CDT, regarding efficiency, firstly, it is relevant to resume the algorithm complexities involved. Then, while CDT is $O(\tau_N \cdot \sigma_N)$, the FWHT shuffling routine is $O(n \cdot \log n)$, or $O(\beta \cdot 2^\beta)$ in terms of the β parameter. Adoption of the CDT strategy as a reference is justified in 1.4, and 1.5. CDT sampling complexity, being governed by τ, the tail-cut factor, and σ, the standard deviation, depends on the size of the sampling interval. FWHT sampling complexity is governed by the β exponent, responsible for the number of samples, the lattice dimension, and other variables.

In the scope of this work, a more efficient algorithm is the one which takes less central processing unit (CPU) cycles to securely complete the noise generation routine for a given parameter setup. Considering the values of β in the interval $4 \leq \beta \leq 20$, and the further settings of this work, the results of efficiency for the FWHT strategy are better than the CDT strategy results. The difference, measured in CPU cycles, tends to be two orders of magnitude, in favor of the FWHT approach, as it is shown in Table 6.

As for the values of $\mu_{\mathfrak{D}}$, which are supposed to follow the μ_N value of zero, that is observed in all of the FWHT tests but is not observed in the CDT tests. This issue, regarding the mean of the Gaussian, is consistent with the relative entropy results, which are generally bigger for CDT tests. Thus, the relative entropy values show that the PMF vector is closer to the algebraic function as the FWHT strategy is used. In the tests performed, the only case of similar Kullback–Leibler values for CDT and FWHT is $\beta = 7$. The values of $\mu_{\mathfrak{D}}$ following those of $\mu_{\mathfrak{D}}$ denote successful exploitation of the CLT. As seen in Section 2.3.4, the theorem converges as $\mu_{\mathfrak{D}}$ approaches $\mu_{\mathcal{N}}$.

Regarding memory, while the CDT strategy requires it for the table it works with, the FWHT strategy does not.

5.2 Security conclusions

Firstly, it is relevant to discuss how the normal distribution of probability makes sampling secure. Frequently observed in nature, normal distributions present higher complexity and often lower randomness if compared to uniform distributions. If nothing is known about a probability distribution, maximum entropy lies in the uniform case. It would not be attractive to cryptographic ends, but if variance σ^2 is specified, which happens in this work, the Gaussian case happens to offer maximum entropy. Such idea is present in Figure 5. Thus, two of the parameters used, β and σ^2, influence Shannon entropy. Regarding resistance to attacks, if properly adjusted, the random number generator presented here provides RLWE cryptosystems with adequate Shannon entropy levels.

Standard deviation is expected to influence the security level of RLWE-based discrete Gaussian samplers, and so it is verified with regards to the $\sigma_{\mathcal{N}}$ input variable, introduced in Section 3.1. Note that $\sigma_{\mathcal{N}}$ is a dispersion metric, and for random number generation purposes, the bigger dispersion value assumes, the better for that matter.

As for resistance against timing attacks, a strengthening isochronous design is implemented in parts of the code. Though, isochrony is not a binary characteristic, and it demands cautious code study, e.g., a given conditional statement often represent a vulnerability to timing attacks, and in that case, an invariant execution time structure can reduce the referred risk. But enforcing isochrony in the referred conditional statement may also lead to severe efficiency damage.

By testing the isochronous timing-resistant FWHT-based DGS scheme in a software implementation running on a simple hardware setup, its feasibility is verified.

5.3 Downsides

Downsides of this work firstly evoke the fact that isochronous DGS constructions are considered computationally costly in lattice-based cryptography. Also, it is relevant to mention the established comparison presented here is built only upon suitable dimensions for Hadamard matrices, that is, 2^{β}. Finally, software tests of this work do not reach high values of β.

5.4 Open questions

This section presents barriers future research is expected to overcome. In the scope of this work, there are both general and specific points needing attention for continued improvement.

Generally considering lattice-based cryptography there are relevant questions that remain open, concerning the rationale of the present work. Sampling from discrete normal distributions is one of those questions, regarding the low-level inner workings of its supporting algorithms, which are based on computationally costly evaluations of integrals and series. Additionally, in terms of key size, e.g., in bits, classical schemes like the RSA system deal with smaller values. As sample sizes of this work are proportional to key sizes, smaller keys could reduce sampling time.

Specifically considering the present work harder scrutiny can be conducted by finding ways to obtain better numbers of $\sigma_{\mathcal{D}}$. Also, cryptology cares about the inverse of functions, thus, it might be worthwhile to analyze the inverse of the FWHT. Once it is considered as an FFT, a computational complexity of $O(n \log n)$ is expected for both the FWHT and for its inverse. But given the many properties involved in an FWHT, proper analyses should be conducted, in order to better understand if the referred properties can affect the asymptotic behavior of its inverse.

REFERENCES

AJTAI, M. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*. New York, NY, USA: ACM, 1996. p. 99–108. ISBN 0-89791-785-5. STOC '96. Available from Internet: <http://doi.acm.org/10.1145/237814.237838>.

AKLEYLEK, S. et al. *qTESLA Git repository*. 2019. Available from Internet: <https://github.com/qtesla/qTesla>. Cited August 28th, 2019.

ALAGIC, G. et al. *Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process*. National Institute of Standards and Technology, 2022. NIST IR 8413-upd1. Available from Internet: <https://doi.org/10.6028/NIST.IR.8413-upd1>. Cited October 21st, 2022.

ALBRECHT, M. R. et al. *Estimate all the LWE, NTRU schemes!* IACR, 2018. Cryptology ePrint Archive, Report 2018/331. Available from Internet: <https://eprint.iacr.org/2018/331>. Cited May 22nd, 2021.

ALBRECHT, M. R. et al. *Estimate all the LWE, NTRU schemes!* 2019. Available from Internet: <https://estimate-all-the-lwe-ntru-schemes.github.io/>. Cited August 29th, 2019.

ALKIM, E. et al. *The Lattice-Based Digital Signature Scheme qTESLA*. IACR, 2019. Cryptology ePrint Archive, Report 2019/085. Available from Internet: <https://eprint.iacr.org/2019/085>. Cited May 22nd, 2021.

ALKIM, E. et al. *FrodoKEM Learning With Errors Key Encapsulation Algorithm Specifications And Supporting Documentation*. 2020. Available from Internet: <https://frodokem.org/files/FrodoKEM-specification-20200930.pdf>. Cited November 26th, 2020.

ANDERSON, R. *Contact Tracing in the Real World*. 2020. Light Blue Touchpaper Security Research, Computer Laboratory, University of Cambridge. Available from Internet: <https://www.lightbluetouchpaper.org/2020/04/12/contact-tracing-in-the-real-world/>. Cited April 19th, 2020.

AONO, Y.; NGUYEN, P. Q. Random sampling revisited: lattice enumeration with discrete pruning. In CORON, J.-S.; NIELSEN, J. B. (Ed.). *Advances in Cryptology –EUROCRYPT 2017*. Switzerland: Springer Nature, 2017. (Lecture Notes in Computer Science), p. 65–102. ISBN 978-3-319-56614-6. Available from Internet: <https://link.springer.com/chapter/10.1007/978-3-319-56614-6_3>. Cited May 6th, 2021.

ARANHA, D. F.; GOUVÊA, C. P. L. *RELIC is an Efficient LIbrary for Cryptography*. 2020. Available from Internet: <https://github.com/relic-toolkit/relic>. Cited April 10th, 2020.

AVANZI, R. et al. *CRYSTALS-KYBER Algorithm Specifications And Supporting Documentation*. CRYSTALS Team, 2021. Available from Internet: <https://www.pq-crystals.org/>. Cited November 2nd, 2021.

BAI, S. et al. *CRYSTALS-Dilithium Algorithm Specifications and Supporting Documentation (Version 3.1)*. CRYSTALS Team, 2021. Available from Internet: <https://www.pq-crystals.org/>. Cited November 2nd, 2021.

BARGUIL, J. M. de M. *EFFICIENT METHODS FOR LATTICE-BASED CRYPTOGRAPHY*. MSc Thesis (MSc) — Universidade de São Paulo, São Paulo, Brasil, OCTOBER 2015. Supervisor: Prof. Dr. Paulo Sérgio L. M. Barreto.

BARRETO, P. S. L. M. et al. A panorama of post-quantum cryptography. In KOç Çetin K. (Ed.). *Open Problems in Mathematics and Computational Science*. Switzerland: Springer, Cham, 2014. p. 387–439. ISBN Online 978-3-319-10683-0. Available from Internet: <https://doi.org/10.1007/978-3-319-10683-0_16>. Cited April 13th, 2020.

BARTHE, G. et al. GALACTICS: Gaussian Sampling for Lattice-Based Constant-Time Implementation of Cryptographic Signatures, Revisited. 2019.

BEER, T. Walsh transforms. *American Journal of Physics*, American Association of Physics Teachers, Vol. 49, no. 5, p. 466–472, May 1981. Am. J. Phys.

BERNSTEIN, D. J. et al. *Post Quantum Cryptography*. 1st. ed. Berlin Heidelberg: Springer Publishing Company, Incorporated, 2008. ISBN 9783540887010.

BERNSTEIN, D. J. et al. *Lattice-based public-key cryptography*. 2021. Post-quantum cryptography. Available from Internet: <http://pqcrypto.org/lattice.html>. Cited April 12th, 2021.

BERRY, A. C. The accuracy of the gaussian approximation to the sum of independent variates. *AMS journal*, AMS, p. 122–136, May 1940. Columbia University.

BHARUCHA-REID, A. T.; SAMBANDHAM, M. Convergence and limit theorems for random polynomials. In BIRNBAUM, Z. W.; LUKACS, E. (Ed.). *Random Polynomials*. Orlando, Florida 32887, United States of America: Academic Press, Inc., 1986, (Probability and mathematical statistics). chap. 8, p. 173–195. ISBN 0-12-095710-8.

BILLINGSLEY, P. The Lindeberg–Lévy theorem for martingales. *Proceedings of the American Mathematical Society*, American Mathematical Society, Providence, Rhode Island, United States of America, vol. 12, p. 788–792, 1961.

BILLINGSLEY, P. *Probability and Measure*. Third edition. New York, NY, United States of America: John Wiley & Sons, Inc., 1995. WILEY SERIES IN PROBABILITY AND MATHEMATICAL STATISTICS. ISBN 0-471-00710-2.

BOAS, P. van E. *ANOTHER NP-COMPLETE PARTITION PROBLEM AND THE COMPLEXITY OF COMPUTING SHORT VECTORS IN A LATTICE*. Mathematisch Instituut, Roetersstraat 15, 1018 WB Amsterdam, The Netherlands, 1981. PREPRINT.

BRUMLEY, D.; BONEH, D. Remote timing attacks are practical. In *Proceedings of the 12th USENIX Security Symposium*. California, USA: USENIX Association, 2003. Available from Internet: <https://www.usenix.org/legacy/publications/library/proceedings/sec03/>. Cited September 26th, 2021.

CHEN, H. *Solving Ring-LWE over Algebraic Integer Rings*. 2019. Cryptology ePrint Archive, Report 2019/791. Available from Internet: <https://eprint.iacr.org/2019/791>.

COMINETTI, E. L. *IMPROVING CLOUD BASED ENCRYPTED DATABASES*. MSc Thesis (MSc) — Universidade de São Paulo, São Paulo, Brasil, 2019. Supervisor: Prof. Dr. Marcos Antonio Simplicio Junior.

CONTE, T. M. et al. Rebooting Computing: The Road Ahead. *Computer*, vol. 50, no. 1, p. 20–29, 2017.

COOLEY, J. W.; TUKEY, J. W. An algorithm for the machine calculation of complex Fourier series. *Math. Comp.*, AMS, American Mathematical Society, 201 Charles Street Providence, Rhode Island, no. 19, p. 297–301, 1965.

COVER, T. M.; THOMAS, J. A. *ELEMENTS OF INFORMATION THEORY*. Second edition. Hoboken, New Jersey: John Wiley & Sons, Inc., 2006. ISBN 9780471241959, 0471241954.

CSRC. Update notice, *NIST Announces Additional Digital Signature Candidates for the PQC Standardization Process*. United States of America: National Institute of Standards and Technology, 2023. Available from Internet: <https://csrc.nist.gov/news/2023/additional-pqc-digital-signature-candidates>. Cited July 20th, 2023.

CSRC, N. *Post-Quantum Cryptography*. 2019. Available from Internet: <https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/>. Cited November 4th, 2022.

DING, J. Post-quantum key exchange based on the LWE and RLWE problems. XIX Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais. 2019.

DUCAS, L. et al. *Lattice Signatures and Bimodal Gaussians*. IACR, 2013. Cryptology ePrint Archive, Report 2013/383. Available from Internet: <https://eprint.iacr.org/2013/383>. Cited March 8th, 2021.

DUCAS, L. et al. *Hawk: Module LIP makes Lattice Signatures Fast, Compact and Simple*. IACR, 2022. Cryptology ePrint Archive, Paper 2022/1155. Available from Internet: <https://eprint.iacr.org/2022/1155>. Cited October 20th, 2022.

DWARAKANATH, N. C.; GALBRAITH, S. D. Sampling from discrete gaussians for lattice-based cryptography on a constrained device. *Applicable Algebra in Engineering, Communications and Computing*, Springer Berlin Heidelberg, vol. 25, no. 3, p. 159–180, 2014. ISSN 0938-1279.

ESPITAU, T. et al. *MITAKA: A Simpler, Parallelizable, Maskable Variant of Falcon*. United States of America: NIST, 2021. Available from Internet: <https://csrc.nist.gov/>. Cited November 9$^{\text{th}}$, 2022.

EVANGELARAS, H.; KOUKOUVINOS, C.; SEBERRY, J. Applications of Hadamard matrices. *JOURNAL OF TELECOMMUNICATIONS AND INFORMATION TECHNOLOGY*, no. 2/2003, p. 3–10, 2003.

FOLLÁTH, J. Gaussian sampling in lattice based cryptography. *Tatra Mountains Mathematical Publications*, De Gruyter, Berlin, Germany, no. 60, p. 1–23, 2014. Mathematical Institute, Slovak Academy of Sciences.

FOUQUE, P.-A. et al. Specification, *Falcon: Fast-Fourier Lattice-based Compact Signatures over NTRU*. 2020. Specification v1.2 –01/10/2020. Available from Internet: <https://falcon-sign.info/>. Cited November 9$^{\text{th}}$, 2022.

GALASSI, M.; THEILER, J. *GNU Scientific Library*. 2019. Available from Internet: <https://www.gnu.org/software/gsl/>. Cited November 22$^{\text{nd}}$, 2019.

GCC Team. Web site, *GCC, the GNU Compiler Collection*. Boston, USA: Free Software Foundation, Inc., 2023. GNU Project. Available from Internet: <https://gcc.gnu.org/>. Cited December 17$^{\text{th}}$, 2023.

GENISE, N. et al. Improved Discrete Gaussian and Subgaussian Analysis for Lattice Cryptography. p. 1–34, March 2020.

GILLIES, D. *Philosophical Theories of Probability*. United Kingdom: Taylor & Francis, 2012. Ebook. ISBN 9781134672455.

GOLDREICH, O.; GOLDWASSER, S.; HALEVI, S. Public-key cryptosystems from lattice reduction problems. MIT, 1997.

GRÄTZER, G. *Lattice Theory: Foundation*. Basel, Switzerland: Birkhäuser, Springer Basel AG, 2011. ISBN 978-3-0348-0018-1.

HARWIT, M.; SLOANE, N. J. A. Appendix Hadamard and S-Matrices, Walsh Functions, Pseudo-Random Sequences, and the Fast Hadamard Transform. In *Hadamard Transform Optics*. Elsevier Inc., 1979. p. 200–228. ISBN 978-0-12-330050-8. Available from Internet: <https://www.sciencedirect.com/book/9780123300508/>. Cited 2021.

HEFFERON, J. *Linear Algebra*. Third edition. Vermont, USA: Saint Michael's College, 2017.

HOFFSTEIN, J.; PIPHER, J.; SILVERMAN, J. H. NTRU: A ring-based public key cryptosystem. In BUHLER, J. P. (Ed.). *Algorithmic Number Theory, Third International Symposium, ANTS-III June 21–25, 1998 Proceedings*. Springer, Berlin, Heidelberg: Springer Nature, 1998. p. 267–288. ISBN 978-3-540-69113-6. Available from Internet: <https://link.springer.com/chapter/10.1007%2FBFb0054868>. Cited February 10th, 2021.

HOFFSTEIN, J.; PIPHER, J.; SILVERMAN, J. H. *US6081597A - Public key cryptosystem method and apparatus, United States Patent, Patent Number 6,081,597*. 2000. United States Patent, Patent Number: 6,081,597. Date of Patent: Jun. 27, 2000.

HOFFSTEIN, J.; PIPHER, J.; SILVERMAN, J. H. *US6081597A - Public key cryptosystem method and apparatus*. 2020. Accessed in April 13th, 2020. Available from Internet: <https://patents.google.com/patent/US6081597>. Cited April 13th, 2020.

HOUSE, P. *Webster's encyclopedic unabridged dictionary of the English language*. New York, USA: Portland House, a division of dilithium Press, Ltd., 1989. ISBN 0-517-68781-X.

HOWE, J. et al. Isochronous Gaussian Sampling: From Inception to Implementation With Applications to the Falcon Signature Scheme. p. 1–23, 2019.

HÜLSING, A. et al. *High-speed key encapsulation from NTRU*. 2017. Cryptology ePrint Archive, Report 2017/667. Available from Internet: <https://eprint.iacr.org/2017/667>.

ION, M. et al. On deploying secure computing commercially: Private intersection-sum protocols and their business applications. IACR, 2019.

JOHANSSON, F. et al. *mpmath: a Python library for arbitrary-precision floating-point arithmetic (version 0.18)*. 2013. Available from Internet: <http://mpmath.org/>. Cited JUNE 14th, 2019.

JÚNIOR, M. B. *Efficient Gaussian sampling for RLWE-based cryptography through a fast Fourier transform*. Porto Alegre, Rio Grande do Sul, Brasil: Sociedade Brasileira de Computação, 2022. 195-208 p. XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. Available from Internet: <https://sol.sbc.org.br/index.php/sbseg/article/view/21668/21492>. Cited September 15th, 2022.

JÚNIOR, M. B.; JUNIOR, M. A. S. Central Limit Theorem Exploitation Optimizes Quantum-Safe Discrete Gaussian Sampling. In SOUZA, S. N. A. de et al. (Ed.). *Anais do VIII Workshop de Pós-Graduação em Engenharia de Computação*. São Paulo/SP, Brasil: Programa de Pós-Graduação em Engenharia Elétrica, 2019. p. 25–28. ISBN 9788553380107. Available from Internet: <https://pcs.usp.br/wpgec/>. Cited June 19th, 2020.

KAHN, D. *The Code Breakers*. Revised and updated. 1230 Avenue of the Americas, New York, NY 10020: SCRIBNER, 1996. ISBN 9780684831305.

KARDAR, M. *II. Probability*. 77 Massachusetts Avenue, Cambridge, MA, USA: [s.n.], 2019. 25–31 p. 8.333: Statistical Mechanics of Particles, Fall 2019. Available from Internet: <http://web.mit.edu/8.333/www/lectures/lec5.pdf>. Cited June 24th, 2021.

KARMAKAR, A. et al. Constant-time discrete gaussian sampling. *IEEE Transactions on Computers*, IEEE, vol. 67, no. 11, November 2018. ISSN Electronic: 1557-9956.

KERRY, C. F.; GALLAGHER, P. D. *Digital Signature Standard (DSS)*. Gaithersburg, MD, United States of America: Information Technology Laboratory, 2013. (Federal Information Processing Standards Publication). FIPS PUB 186-4. Available from Internet: <http://dx.doi.org/10.6028/NIST.FIPS.186-4>. Cited November 2nd, 2022.

KIMBALL, K. *Announcing Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms*. College Park, MD, United States of America: National Archives, 2016. Document Number: 2016-30615. Available from Internet: <https://www.federalregister.gov/d/2016-30615>. Cited November 4th, 2022.

KNUTH, D. E.; YAO, A. C. The Complexity of Nonuniform Random Number Generation. In TRAUB, J. F. (Ed.). *Algorithms and Complexity - NEW DIRECTIONS AND RECENT RESULTS*. 111 Fifth Avenue, New York, New York 10003: ACADEMIC PRESS, INC., 1976. p. 357–428. ISBN 0-12-697540-X. Carnegie-Mellon University.

KOBLITZ, N. Elliptic curve cryptosystems. *MATHEMATICS OF COMPUTATION*, American Mathematical Society, vol. 48, no. 177, p. 203–209, January 1987.

KODUKULA, I.; PINGALI, K. Transformations for imperfectly nested loops. In *Proceedings of the 1996 ACM/IEEE Conference on Supercomputing*. 1730 Massachusetts Ave., NW Washington, DC, United States: IEEE Computer Society, 1996. p. 12–es. ISBN 0897918541. Supercomputing '96. Available from Internet: <https://doi.org/10.1145/369028.369051>. Cited July 27th, 2021.

LENSTRA, A. K.; JR., H. W. L.; LOVÁSZ, L. *Factoring polynomials with rational coefficients*. Switzerland: Springer Nature, 1982. 515–534 p. . Available from Internet: <https://doi.org/10.1007/BF01457454>. Cited June 4th, 2021.

LINDNER, R. et al. *TU Darmstadt Lattice Challenge*. Darmstadt, Germany: [s.n.], 2021. Technische Universität Darmstadt, Department of Computer Science. Available from Internet: <https://www.latticechallenge.org/>. Cited May 6th, 2021.

LOU, X. et al. *A Survey of Microarchitectural Side-channel Vulnerabilities, Attacks, and Defenses in Cryptography*. 1601 Broadway, 10th Floor, New York, NY 10019-7434: ACM, 2021.

LU, Y. et al. Faster Ridge Regression via the Subsampled Randomized Hadamard Transform. University of Pennsylvania, Philadelphia, PA 19104, 2013. Available from Internet: <https://repository.upenn.edu/statistics_papers/221/>. Cited March 8th, 2021.

LYUBASHEVSKY, V. et al. SWIFFT: A Modest Proposal for FFT Hashing. Springer, p. 54–72, 2008. FSE 2008. Available from Internet: <https://iacr.org/archive/fse2008/50860052/50860052.pdf>. Cited August 4th, 2023.

LYUBASHEVSKY, V.; PEIKERT, C.; REGEV, O. A Toolkit for Ring-LWE Cryptography. IACR, May 2013. Available from Internet: <https://eprint.iacr.org/2013/293>. Cited March 8th, 2021.

LYUBASHEVSKY, V.; PEIKERT, C.; REGEV, O. On ideal lattices and learning with errors over rings. *J. ACM*, ACM, New York, NY, USA, vol. 60, no. 6, p. 43:1–43:35, November 2013. ISSN 0004-5411. Available from Internet: <http://doi.acm.org/10.1145/2535925>. Cited March 8th, 2021.

MARGI, C. B. et al. Segurança em redes de sensores sem fio. In SANTIN, A.; NUNES, R. C.; DAHAB, R. (Ed.). *Minicursos: SBSEG 2009 / IX Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*. Porto Alegre, Rio Grande do Sul, Brasil: Sociedade Brasileira de Computação, 2009. vol. 1, p. 149–194.

MARINGER, G.; FRITZMANN, T.; SEPÚLVEDA, J. *The Influence of LWE/RLWE Parameters on the Stochastic Dependence of Decryption Failures*. 2019. Available from Internet: <https://eprint.iacr.org/2019/1469>. Cited December 31st, 2019.

MENDONÇA, J. R. G. de. Audiovisual, *Estatística e Probabilidade - O teorema central do limite*. YouTube, 2022. UNIVESP. Available from Internet: <https://www.youtube.com/watch?v=34qb9m0NeNc>. Cited November 16th, 2022.

MICCIANCIO, D. *Sampling, Lattice Cryptography*. 2019. Available from Internet: <http://cseweb.ucsd.edu/~daniele/LatticeLinks/Sampling.html>. Cited June 6th, 2019.

MICCIANCIO, D.; WALTER, M. Gaussian sampling over the integers: Efficient, generic, constant-time. IACR, p. 01–28, February 2018.

MOODY, D. *NIST PQC: looking into the future*. United States of America: National Institute of Standards and Technology, 2022. NIST. Available from Internet: <https://csrc.nist.gov/csrc/media/Presentations/2022/nist-pqc-looking-into-the-future/images-media/session-1-moody-looking-into-future-pqc2022.pdf>. Cited January 23rd, 2023.

MORAIS, M. *Lecture 8: Information Theory and Maximum Entropy*. 2018. 1-7 p. NEU 560: Statistical Modeling and Analysis of Neural Data, Spring 2018. Available from Internet: <http://pillowlab.princeton.edu/teaching/statneuro2018/slides/notes08_infotheory.pdf>. Cited May 26th, 2021.

NEUMANN, J. von. *Various techniques used in connection with random digits*. 1951. 36–38 p. National Bureau of Standards Applied Mathematics Series.

NEYMAN, J. Outline of a Theory of Statistical Estimation Based on the Classical Theory of Probability. *PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A, MATHEMATICAL, PHYSICAL AND ENGINEERING SCIENCES*, THE ROYAL SOCIETY PUBLISHING, London, SW1, United Kingdom, vol. 236, no. 767, August 1937. ISSN 2054-0272.

NIST. Campaign audiovisual, *Post-Quantum Cryptography: the Good, the Bad, and the Powerful*. United States of America: YouTube, 2021. Available from Internet: <https://www.youtube.com/watch?v=uE_Y1C4QPU8>. Cited November 13th, 2022.

ODLYZKO, A. M. The rise and fall of knapsack cryptosystems. AT& T Bell Laboratories, AT& T Bell Laboratories, Murray Hill, New Jersey 07974, USA, 1990.

ORSINI, L. de Q. *Curso de Circuitos Elétricos*. São Paulo - SP - Brasil: Editora Edgard Blücher Ltda., 1994. vol. 2.

ORTIZ, J. N. *Amostragem Gaussiana aplicada à Criptografia Baseada em Reticulados*. MSc Thesis (MSc) — Universidade Estadual de Campinas, Campinas, Brasil, 2016. Orientador: Prof. Dr. Ricardo Dahab.

ORTIZ, J. N.; ARANHA, D. F.; DAHAB, R. Implementação em tempo constante de amostragem de gaussianas discretas. In MAZIERO, C.; TERADA, R. (Ed.). Porto Alegre, Rio Grande do Sul, Brazil: Sociedade Brasileira de Computação, 2015. p. 239–252. SBSeg 2015. Available from Internet: <https://000626cf-7296-4b40-ae6b-d1a550c81174.usrfiles.com/ugd/000626_5dc702ae45b54dd787adab7de7a7a385.pdf>. Cited June 1st, 2022.

PEIKERT, C. Lattice Cryptography for the Internet. In MOSCA, M. (Ed.). *Post-Quantum Cryptography 6th International Workshop*. Switzerland: Springer International Publishing, 2014. (Lecture Notes in Computer Science), p. 197–219. ISBN 978-3-319-11658-7.

PEIKERT, C. A decade of lattice cryptography. *Found. Trends Theor. Comput. Sci.*, Now Publishers Inc., Hanover, MA, USA, vol. 10, no. 4, p. 283–424, March 2016. ISSN 1551-305X. Available from Internet: <http://dx.doi.org/10.1561/0400000074>. Cited March 8th, 2021.

PEIKERT, C.; PEPIN, Z. Algebraically structured LWE, revisited. JULY 2019.

PERSICHETTI, E. et al. *PQC WIKI, A PLATFORM FOR NIST POST-QUANTUM CRYPTOGRAPHY STANDARDIZATION*. 777 Glades Road, Boca Raton, FL 33431: Florida Atlantic University, 2021. DEPARTMENT OF MATHEMATICAL SCIENCES. Available from Internet: <https://pqc-wiki.fau.edu/>. Cited March 9th, 2021.

PÖPPELMAN, T. et al. *NewHope Algorithm Specifications and Supporting Documentation*. Version 1.03. c/o Thomas Pöppelmann, Infineon Technologies AG, Am Campeon 1-12, 85579 Neubiberg, Germany, 2019. NIST.

POPPER, K. R. *The logic of scientific discovery*. Revised. Oxfordshire, England: Routledge, 2005. ISBN 1-1344-7002-9.

PRATT, W. K.; KANE, J.; ANDREWS, H. C. Hadamard Transform Image Coding. In *PROCEEDINGS OF THE IEEE, VOL. 57, NO. 1*. New York, NY 10016-5997 USA: IEEE, 1969. vol. 57, no. 1, p. 58–68.

RADER, C. M. *A NEW METHOD OF GENERATING GAUSSIAN RANDOM VARIABLES BY COMPUTER*. Lexington, Massachusetts, 1969.

REGEV, O. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, ACM, New York, NY, USA, vol. 56, no. 6, p. 34:1–34:40, September 2009. ISSN 0004-5411. Available from Internet: <http://doi.acm.org/10.1145/1568318.1568324>. Cited March 8th, 2021.

REICHERT, L.; BRACK, S.; SCHEUERMANN, B. *Privacy-Preserving Contact Tracing of COVID-19 Patients*. 2020. Cryptology ePrint Archive, Report 2020/375. Available from Internet: <https://eprint.iacr.org/2020/375>. Cited March 8th, 2021.

REPARAZ, O.; BALASCH, J.; VERBAUWHEDE, I. *Dude, is my code constant time?* 2016. Cryptology ePrint Archive, Report 2016/1123. Available from Internet: <https://eprint.iacr.org/2016/1123>.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, ACM, New York, NY, USA, vol. 21, no. 2, p. 120–126, FEBRUARY 1978. ISSN 0001-0782. Available from Internet: <http://doi.acm.org/10.1145/359340.359342>. Cited March 8th, 2021.

ROY, S. S.; VERCAUTEREN, F.; VERBAUWHEDE, I. High Precision Discrete Gaussian Sampling on FPGAs. In LANGE, T.; LAUTER, K.; LISONĚK, P. (Ed.). *Selected Areas in Cryptography – SAC 2013*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. p. 383–401. ISBN 978-3-662-43414-7.

SAARINEN, M.-J. O. *Gaussian sampling precision in lattice cryptography*. 2015. Cryptology ePrint Archive, Report 2015/953. Available from Internet: <https://eprint.iacr.org/2015/953>. Cited May 26th, 2021.

SAKAMOTO, J.; MORI, Y.; SEKIOKA, T. Probability analysis method using Fast Fourier transform and its application. *Structural Safety*, Elsevier Science Ltd., The Netherlands, vol. 19, no. 1, p. 21–36, 1997. PII: S0167-4730(96)00032-X.

SCHNORR, C. P. Lattice Reduction by Random Sampling and Birthday Methods. *LNCS*, Springer-Verlag, Germany, no. 2607, p. 145–156, 2003. STACS 2003.

SCHNORR, C. P.; EUCHNER, M. *Lattice basis reduction: improved practical algorithms and solving subset sum problems*. Switzerland: Springer Nature, 1994. 181–199 p. The Mathematical Programming Society, Inc. Available from Internet: <https://doi.org/10.1007/BF01581144>. Cited June 3rd, 2021.

SHOR, P. W. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 3 Park Avenue, 17th Floor, New York, NY 10016-5997 USA: IEEE, 1994. p. 124–134. ISBN 0-8186-6580-7. Available from Internet: <https://ieeexplore.ieee.org/document/365700/>. Cited September 13th, 2021.

SHOR, P. W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. arXiv, AT& T Research, Room 2D-149, 600 Mountain Ave., Murray Hill, NJ 07974, January 1996. Original article published in 1995.

SHOUP, V. *A Computational Introduction to Number Theory and Algebra*. Version 2. Cambridge CB2 8RU, UK: Cambridge University Press, 2008.

STEBILA, D.; MOSCA, M. *Open Quantum Safe*. 2021. Available from Internet: <https://openquantumsafe.org/>. Cited February 21st, 2021.

SUMMIT, S. *comp.lang.c FAQ list · Question 20.14*. 2019. Available from Internet: <http://www.c-faq.com/misc/eff2.html>. Cited December 19th, 2019.

SUN, S. et al. *Generic, efficient and isochronous Gaussian sampling over the integers*. 2021. Cryptology ePrint Archive, Report 2021/199. Available from Internet: <https://eprint.iacr.org/2021/199>. Cited April 12th, 2021.

SYLVESTER, J. J. Lx. thoughts on inverse orthogonal matrices, simultaneous sign-successions, and tessellated pavements in two or more colours, with applications to newton's rule, ornamental tile-work, and the theory of numbers. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, Taylor & Francis, vol. 34, no. 232, p. 461–475, 1867. Available from Internet: <https://doi.org/10.1080/14786446708639914>. Cited 2021.

Technical Committee: ISO/IEC JTC 1/SC 22. Standard, *ISO/IEC 9899:1999, Programming languages, C*. Geneva, Switzerland: International Organization for Standardization, 1999. (Series: ISO/IEC 9899). Status: Withdrawn.

TERADA, R. *Segurança de Dados*. 2ª edição. ed. Brasil: Editora Blucher, 2008. ISBN 9788521204398.

TERUYA, T.; KASHIWABARA, K.; HANAOKA, G. *Fast Lattice Basis Reduction Suitable for Massive Parallelization and Its Application to the Shortest Vector Problem*. 2018. Cryptology ePrint Archive, Report 2018/044. Available from Internet: <https://eprint.iacr.org/2018/044>. Cited MAY 6th, 2021.

TSCHOFENIG, H.; BACCELLI, E. Cyberphysical security for the masses a survey of the internet protocol suite for internet of things security. IEEE, IEEE Headquarters, Three Park Ave., 17th Floor, New York, NY 10016-5997, vol. 56, no. 5, p. 47–57, September/October 2019. ISSN 1540-7993.

WANG, J.; LING, C. Polar sampler: Discrete gaussian sampling over the integers using polar codes. 2019.

WANG, Z. Markov chain monte carlo methods for lattice gaussian sampling: Convergence analysis and enhancement. *IEEE Transactions on Communications*, IEEE, Early Access, no. Early Access, July 2019. ISSN Electronic ISSN: 1558-0857. Available from Internet: <https://ieeexplore.ieee.org/document/8753603>. Cited March 8th, 2021.

WANG, Z.; LING, C. Lattice gaussian sampling by markov chain monte carlo: Bounded distance decoding and trapdoor sampling. *IEEE Transactions on Information Theory*, IEEE, vol. 65, no. 6, June 2019. ISSN Electronic: 1557-9654. Available from Internet: <https://ieeexplore.ieee.org/document/8653323>. Cited March 8th, 2021.

WANG, Z.; LYU, S.; LIU, L. Learnable markov chain monte carlo sampling methods for lattice gaussian distribution. *IEEE Access*, IEEE, vol. 7, June 2019. ISSN Electronic: 2169-3536. Available from Internet: <https://ieeexplore.ieee.org/document/8747498>. Cited March 8th, 2021.

WEHRUNG, F. et al. *Lattice Theory: Special Topics and Applications, Volume 2*. Switzerland: Birkhäuser, Springer International Publishing AG, 2016. ISBN 978-3-319-44236-5 (eBook).

WEISSTEIN, E. W. *Probability Density Function*. 2021. From MathWorld–A Wolfram Web Resource. Available from Internet: <https://mathworld.wolfram.com/ProbabilityDensityFunction.html>. Cited April 4th, 2021.

WHYTE, W. et al. *IEEE Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices*. 2008. IEEE Std 1363.1-2008.

ZHAO, R. K.; STEINFELD, R.; SAKZAD, A. *COSAC: COmpact and Scalable Arbitrary-Centered Discrete Gaussian Sampling over Integers*. 2019. Cryptology ePrint Archive, Report 2019/1011. Available from Internet: <https://eprint.iacr.org/2019/1011>. Cited April 12th, 2021.

ZHAO, R. K.; STEINFELD, R.; SAKZAD, A. FACCT: FAst, Compact, and Constant-Time Discrete Gaussian Sampler over Integers. *IEEE TRANSACTIONS ON COMPUTERS*, IEEE, IEEE Headquarters, Three Park Ave., 17th Floor, New York, NY 10016-5997, vol. 69, no. NO. 1, p. 126–137, January 2020.

APPENDIX A - TABLES AND SUPPLEMENTARY INFORMATION

The first section of this appendix presents a short description of the NIST standardization process for post-quantum cryptography. The following section presents a simple discretization example for a normal probability distribution. Then, there is a section showing the complete tables containing values of metrics, and Gaussian sampling samples for both FWHT and CDT sampling strategies, as regarded in this work.

A.1 The NIST standardization process

The National Institute of Standards and Technology (NIST) is a federal agency of the United States of America government. They have successfully conducted the creation of notable cryptography standards such as the SHA family of hash functions, and the AES block cipher (TERADA, 2008).

Since 2016, the NIST is officially conducting another standardization process. This time they are working on post-quantum cryptographic schemes, which are expected to offer resistance against classical and quantum computing attacks. Supposed to be concluded in the year of 2024, the current process evaluates schemes covering primitives for digital signing, key exchange, and public-key encryption. Candidate proposals should comply with up to five security levels, according to Table 7, and the evaluation criteria is mainly based on security and performance. The standardization process is formatted as a competition, divided into a number of stages or rounds. An open public call was initially announced by the NIST, and since then, interested parties started submitting their candidate schemes, firstly for public scrutiny, and then for the NIST evaluation itself (KIMBALL, 2016; CSRC, 2019; NIST, 2021; MOODY, 2022).

Table 7: The security levels.

Level	Minimal hardness
I	AES-128 against an exhaustive key search attack.
II	SHA-256 against a collision search attack.
III	AES-192 against an exhaustive key search attack.
IV	SHA-384 against a collision search attack.
V	AES-256 against an exhaustive key search attack.

Source: (MOODY, 2022).

In Table 7, the Level column presents the five possible security levels a given candidate scheme can address. Column Minimal hardness describes each security level. If a given candidate scheme claims to offer a security level I parameter set, then by using the referred parameter set, the scheme should be at least as hard to break as AES-128 against an exhaustive key search attack. A candidate scheme can support more than one security levels, e.g., the already selected scheme know as CRYSTALS-Dilithium supports levels II, III and V.

By the end of a given competition stage, some candidates are declared ineligible for standardization, a few may be considered good enough, being early picked for standardization, while other schemes qualify for the next stage. Three rounds were initially planned but additional ones proved necessary (ALAGIC et al., 2022). In July 2023, NIST announced additional digital signature candidates to be considered in the process (CSRC, 2023).

An independent annotated compendium on the standardization process evolution is available in (PERSICHETTI et al., 2021).

A.2 Discretization of a continuous Gaussian

This appendix section presents the basics for the practical process of turning a continuous-case normal distribution probability density function into its discrete-case counterpart. Bounds for the PDF are defined by the [-183, 183] interval limits. By using the notation of Equation 2.9, a continuous-case algebraic Gaussian function is $\mathcal{N}(0,\ 216.391560416985)$ is used. It denotes a $\sigma_{\mathcal{N}}$ standard deviation value of 14.7102535809885, and a $\mu_{\mathcal{N}}$ mean value of zero. This test is executed with a Python language routine, supported by the mpmath library (JOHANSSON et al., 2013). Initially, a seed value of 31415926536 is defined, and the routine randomly acquires 2,048 observation values from the bounded PDF, in order to form a sample. Table 8 presents earlier results obtained with DiGS. An Obs. header title denotes the observation position in the sample vector, and a Val. header title denotes the value observed. Corresponding plots for the histogram, the approximate function made of line segments, and the approximate normal PDF are presented next, in Figure 15. The approximate Gaussian is loosely related to the algebraic one. It has a $\sigma_{\mathfrak{D}}$ value of 1.33863166279, and a $\mu_{\mathfrak{D}}$ mean value of -178.668457031 for which the maximum probability of 0.445 occurs. Nevertheless, for this work, it represents the beginning of successful Gaussian fitting routines. The routine completes in $2.206545996669 \cdot 10^9$ CPU cycles, and the entropy obtained for $\sigma_{\mathfrak{D}}$ is 7.62459100595 bits.

Table 8: Python calibration-testing samples, using the mpmath library.

Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.
0	-178	256	-183	512	-183	768	-181	1024	-178	1280	-179	1536	-179	1792	-180
1	-178	257	-177	513	-178	769	-181	1025	-181	1281	-178	1537	-177	1793	-178
2	-182	258	-181	514	-179	770	-179	1026	-179	1282	-180	1538	-180	1794	-179
3	-181	259	-183	515	-177	771	-178	1027	-178	1283	-178	1539	-180	1795	-179
4	-177	260	-181	516	-178	772	-180	1028	-179	1284	-178	1540	-179	1796	-181
5	-178	261	-180	517	-179	773	-180	1029	-178	1285	-183	1541	-179	1797	-181

Table 8 continuation from previous page.

Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.
6	-179	262	-178	518	-178	774	-179	1030	-178	1286	-178	1542	-183	1798	-177
7	-179	263	-181	519	-183	775	-180	1031	-182	1287	-179	1543	-179	1799	-181
8	-178	264	-179	520	-180	776	-178	1032	-178	1288	-178	1544	-177	1800	-179
9	-178	265	-178	521	-177	777	-178	1033	-178	1289	-181	1545	-178	1801	-178
10	-182	266	-178	522	-178	778	-181	1034	-179	1290	-179	1546	-179	1802	-180
11	-177	267	-178	523	-178	779	-178	1035	-182	1291	-179	1547	-178	1803	-183
12	-177	268	-179	524	-178	780	-178	1036	-180	1292	-180	1548	-178	1804	-178
13	-178	269	-180	525	-179	781	-179	1037	-179	1293	-178	1549	-178	1805	-179
14	-178	270	-180	526	-178	782	-179	1038	-178	1294	-180	1550	-180	1806	-181
15	-178	271	-178	527	-179	783	-178	1039	-178	1295	-178	1551	-179	1807	-178
16	-178	272	-178	528	-180	784	-177	1040	-179	1296	-183	1552	-180	1808	-179
17	-178	273	-182	529	-183	785	-178	1041	-178	1297	-179	1553	-180	1809	-178
18	-181	274	-177	530	-179	786	-178	1042	-178	1298	-178	1554	-178	1810	-180
19	-180	275	-179	531	-177	787	-183	1043	-181	1299	-179	1555	-178	1811	-180
20	-178	276	-178	532	-179	788	-177	1044	-179	1300	-178	1556	-179	1812	-178
21	-177	277	-178	533	-181	789	-178	1045	-178	1301	-178	1557	-177	1813	-179
22	-177	278	-177	534	-183	790	-178	1046	-178	1302	-179	1558	-178	1814	-178
23	-179	279	-178	535	-177	791	-179	1047	-179	1303	-178	1559	-179	1815	-180
24	-178	280	-179	536	-178	792	-178	1048	-180	1304	-179	1560	-178	1816	-181
25	-178	281	-177	537	-178	793	-178	1049	-180	1305	-178	1561	-178	1817	-178
26	-180	282	-179	538	-178	794	-178	1050	-178	1306	-178	1562	-177	1818	-180
27	-178	283	-181	539	-179	795	-177	1051	-180	1307	-179	1563	-178	1819	-177
28	-178	284	-177	540	-178	796	-178	1052	-180	1308	-178	1564	-178	1820	-178
29	-178	285	-179	541	-178	797	-179	1053	-178	1309	-179	1565	-178	1821	-180
30	-181	286	-178	542	-179	798	-182	1054	-179	1310	-177	1566	-178	1822	-178
31	-178	287	-178	543	-178	799	-179	1055	-178	1311	-177	1567	-178	1823	-180
32	-179	288	-177	544	-181	800	-177	1056	-180	1312	-178	1568	-178	1824	-178
33	-180	289	-179	545	-179	801	-177	1057	-179	1313	-179	1569	-178	1825	-178
34	-178	290	-180	546	-178	802	-179	1058	-178	1314	-178	1570	-177	1826	-177
35	-179	291	-180	547	-180	803	-177	1059	-178	1315	-179	1571	-178	1827	-182
36	-178	292	-181	548	-179	804	-178	1060	-178	1316	-178	1572	-178	1828	-178
37	-178	293	-177	549	-178	805	-179	1061	-180	1317	-178	1573	-178	1829	-178
38	-177	294	-178	550	-179	806	-179	1062	-179	1318	-178	1574	-179	1830	-177
39	-179	295	-178	551	-178	807	-179	1063	-178	1319	-178	1575	-177	1831	-178
40	-179	296	-180	552	-178	808	-179	1064	-177	1320	-178	1576	-182	1832	-180
41	-178	297	-177	553	-179	809	-178	1065	-177	1321	-177	1577	-179	1833	-179
42	-179	298	-177	554	-178	810	-178	1066	-181	1322	-177	1578	-178	1834	-178
43	-179	299	-179	555	-181	811	-180	1067	-178	1323	-178	1579	-179	1835	-179
44	-179	300	-178	556	-178	812	-178	1068	-178	1324	-178	1580	-178	1836	-180

Table 8 continuation from previous page.

Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.
45	-178	301	-179	557	-178	813	-180	1069	-177	1325	-178	1581	-179	1837	-180
46	-179	302	-180	558	-179	814	-178	1070	-180	1326	-180	1582	-178	1838	-178
47	-179	303	-177	559	-178	815	-179	1071	-179	1327	-178	1583	-183	1839	-179
48	-178	304	-178	560	-178	816	-178	1072	-178	1328	-179	1584	-178	1840	-179
49	-179	305	-177	561	-177	817	-178	1073	-179	1329	-179	1585	-181	1841	-177
50	-178	306	-181	562	-178	818	-178	1074	-179	1330	-179	1586	-178	1842	-178
51	-178	307	-178	563	-180	819	-179	1075	-177	1331	-178	1587	-178	1843	-182
52	-178	308	-179	564	-180	820	-179	1076	-178	1332	-178	1588	-181	1844	-177
53	-178	309	-179	565	-178	821	-179	1077	-178	1333	-179	1589	-178	1845	-177
54	-179	310	-178	566	-178	822	-178	1078	-180	1334	-177	1590	-179	1846	-178
55	-179	311	-178	567	-178	823	-183	1079	-178	1335	-179	1591	-178	1847	-179
56	-178	312	-180	568	-178	824	-178	1080	-181	1336	-178	1592	-178	1848	-177
57	-178	313	-178	569	-179	825	-179	1081	-178	1337	-179	1593	-179	1849	-178
58	-177	314	-181	570	-177	826	-177	1082	-177	1338	-178	1594	-179	1850	-178
59	-181	315	-179	571	-181	827	-178	1083	-178	1339	-179	1595	-178	1851	-178
60	-177	316	-179	572	-177	828	-180	1084	-180	1340	-178	1596	-179	1852	-179
61	-178	317	-178	573	-179	829	-180	1085	-178	1341	-177	1597	-178	1853	-177
62	-179	318	-178	574	-179	830	-177	1086	-178	1342	-181	1598	-182	1854	-178
63	-181	319	-178	575	-179	831	-178	1087	-178	1343	-182	1599	-178	1855	-178
64	-178	320	-178	576	-178	832	-178	1088	-178	1344	-178	1600	-177	1856	-180
65	-178	321	-178	577	-178	833	-182	1089	-178	1345	-178	1601	-179	1857	-178
66	-178	322	-178	578	-181	834	-178	1090	-178	1346	-178	1602	-178	1858	-180
67	-178	323	-178	579	-178	835	-178	1091	-178	1347	-180	1603	-178	1859	-179
68	-178	324	-177	580	-180	836	-179	1092	-178	1348	-177	1604	-178	1860	-177
69	-178	325	-179	581	-178	837	-183	1093	-178	1349	-179	1605	-178	1861	-178
70	-179	326	-178	582	-181	838	-177	1094	-178	1350	-178	1606	-177	1862	-178
71	-178	327	-180	583	-178	839	-178	1095	-178	1351	-178	1607	-177	1863	-177
72	-177	328	-178	584	-177	840	-177	1096	-178	1352	-178	1608	-180	1864	-177
73	-178	329	-179	585	-178	841	-180	1097	-178	1353	-177	1609	-178	1865	-180
74	-178	330	-178	586	-179	842	-177	1098	-178	1354	-179	1610	-183	1866	-178
75	-178	331	-178	587	-179	843	-178	1099	-180	1355	-179	1611	-178	1867	-178
76	-178	332	-178	588	-179	844	-178	1100	-183	1356	-178	1612	-178	1868	-180
77	-178	333	-178	589	-178	845	-181	1101	-178	1357	-179	1613	-179	1869	-179
78	-178	334	-179	590	-178	846	-183	1102	-178	1358	-178	1614	-178	1870	-181
79	-179	335	-178	591	-178	847	-180	1103	-177	1359	-178	1615	-180	1871	-181
80	-179	336	-179	592	-178	848	-178	1104	-178	1360	-178	1616	-177	1872	-179
81	-178	337	-178	593	-180	849	-179	1105	-179	1361	-178	1617	-180	1873	-180
82	-178	338	-178	594	-179	850	-178	1106	-177	1362	-178	1618	-178	1874	-180
83	-180	339	-178	595	-177	851	-178	1107	-178	1363	-182	1619	-179	1875	-178

Table 8 continuation from previous page.

Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.
84	-179	340	-178	596	-180	852	-179	1108	-179	1364	-178	1620	-179	1876	-182
85	-178	341	-182	597	-178	853	-181	1109	-178	1365	-179	1621	-180	1877	-180
86	-178	342	-179	598	-180	854	-178	1110	-178	1366	-178	1622	-178	1878	-178
87	-178	343	-178	599	-178	855	-178	1111	-180	1367	-178	1623	-177	1879	-178
88	-180	344	-178	600	-181	856	-182	1112	-178	1368	-178	1624	-179	1880	-177
89	-178	345	-179	601	-178	857	-179	1113	-183	1369	-178	1625	-178	1881	-179
90	-178	346	-180	602	-178	858	-180	1114	-181	1370	-178	1626	-179	1882	-178
91	-182	347	-178	603	-178	859	-178	1115	-178	1371	-178	1627	-179	1883	-177
92	-179	348	-180	604	-179	860	-178	1116	-178	1372	-178	1628	-178	1884	-180
93	-178	349	-179	605	-178	861	-178	1117	-179	1373	-177	1629	-178	1885	-179
94	-178	350	-178	606	-178	862	-181	1118	-182	1374	-178	1630	-178	1886	-178
95	-178	351	-178	607	-178	863	-181	1119	-177	1375	-178	1631	-178	1887	-178
96	-180	352	-179	608	-177	864	-178	1120	-179	1376	-178	1632	-179	1888	-179
97	-177	353	-179	609	-178	865	-179	1121	-179	1377	-177	1633	-178	1889	-179
98	-178	354	-178	610	-180	866	-179	1122	-178	1378	-178	1634	-179	1890	-181
99	-178	355	-180	611	-178	867	-180	1123	-177	1379	-177	1635	-178	1891	-178
100	-178	356	-178	612	-177	868	-177	1124	-178	1380	-179	1636	-179	1892	-178
101	-180	357	-178	613	-177	869	-179	1125	-178	1381	-177	1637	-178	1893	-177
102	-177	358	-178	614	-177	870	-178	1126	-178	1382	-177	1638	-180	1894	-180
103	-179	359	-177	615	-178	871	-180	1127	-178	1383	-182	1639	-178	1895	-178
104	-180	360	-178	616	-180	872	-179	1128	-178	1384	-178	1640	-178	1896	-178
105	-179	361	-180	617	-182	873	-178	1129	-177	1385	-179	1641	-178	1897	-180
106	-183	362	-180	618	-180	874	-178	1130	-180	1386	-179	1642	-178	1898	-178
107	-181	363	-181	619	-179	875	-180	1131	-178	1387	-177	1643	-183	1899	-178
108	-178	364	-178	620	-178	876	-178	1132	-178	1388	-179	1644	-177	1900	-178
109	-180	365	-179	621	-179	877	-179	1133	-178	1389	-178	1645	-177	1901	-178
110	-178	366	-178	622	-179	878	-179	1134	-179	1390	-178	1646	-179	1902	-179
111	-178	367	-177	623	-179	879	-178	1135	-182	1391	-179	1647	-178	1903	-182
112	-178	368	-177	624	-177	880	-178	1136	-178	1392	-179	1648	-178	1904	-181
113	-180	369	-179	625	-179	881	-179	1137	-177	1393	-178	1649	-178	1905	-180
114	-178	370	-178	626	-177	882	-178	1138	-180	1394	-177	1650	-180	1906	-178
115	-180	371	-180	627	-177	883	-179	1139	-178	1395	-178	1651	-178	1907	-178
116	-178	372	-179	628	-179	884	-177	1140	-179	1396	-178	1652	-178	1908	-180
117	-178	373	-178	629	-178	885	-179	1141	-178	1397	-179	1653	-180	1909	-178
118	-177	374	-183	630	-178	886	-179	1142	-181	1398	-178	1654	-180	1910	-178
119	-179	375	-179	631	-178	887	-179	1143	-180	1399	-177	1655	-180	1911	-178
120	-181	376	-183	632	-179	888	-178	1144	-181	1400	-179	1656	-179	1912	-178
121	-178	377	-179	633	-179	889	-179	1145	-178	1401	-179	1657	-179	1913	-179
122	-177	378	-178	634	-179	890	-181	1146	-177	1402	-179	1658	-180	1914	-180

Table 8 continuation from previous page.

Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.
123	-177	379	-180	635	-178	891	-179	1147	-179	1403	-180	1659	-180	1915	-178
124	-179	380	-179	636	-178	892	-179	1148	-177	1404	-177	1660	-181	1916	-178
125	-181	381	-178	637	-183	893	-178	1149	-178	1405	-177	1661	-181	1917	-177
126	-177	382	-179	638	-177	894	-178	1150	-178	1406	-179	1662	-178	1918	-179
127	-178	383	-178	639	-180	895	-178	1151	-178	1407	-178	1663	-177	1919	-178
128	-179	384	-178	640	-180	896	-177	1152	-180	1408	-180	1664	-178	1920	-180
129	-177	385	-179	641	-178	897	-179	1153	-178	1409	-180	1665	-179	1921	-177
130	-179	386	-177	642	-180	898	-179	1154	-178	1410	-178	1666	-178	1922	-178
131	-179	387	-182	643	-181	899	-178	1155	-178	1411	-178	1667	-180	1923	-178
132	-178	388	-179	644	-178	900	-177	1156	-178	1412	-180	1668	-178	1924	-179
133	-182	389	-179	645	-178	901	-178	1157	-177	1413	-181	1669	-178	1925	-180
134	-177	390	-178	646	-179	902	-178	1158	-179	1414	-178	1670	-177	1926	-178
135	-179	391	-178	647	-182	903	-178	1159	-180	1415	-177	1671	-179	1927	-177
136	-178	392	-180	648	-179	904	-180	1160	-177	1416	-177	1672	-181	1928	-179
137	-178	393	-177	649	-179	905	-177	1161	-178	1417	-178	1673	-178	1929	-177
138	-177	394	-178	650	-179	906	-178	1162	-183	1418	-178	1674	-178	1930	-177
139	-178	395	-179	651	-178	907	-180	1163	-181	1419	-178	1675	-178	1931	-178
140	-178	396	-179	652	-178	908	-179	1164	-179	1420	-180	1676	-178	1932	-181
141	-179	397	-178	653	-182	909	-182	1165	-178	1421	-180	1677	-177	1933	-183
142	-183	398	-178	654	-178	910	-180	1166	-178	1422	-178	1678	-178	1934	-178
143	-180	399	-178	655	-180	911	-178	1167	-178	1423	-179	1679	-178	1935	-181
144	-183	400	-181	656	-177	912	-181	1168	-181	1424	-181	1680	-180	1936	-177
145	-178	401	-180	657	-178	913	-179	1169	-178	1425	-178	1681	-179	1937	-179
146	-178	402	-180	658	-181	914	-179	1170	-178	1426	-177	1682	-178	1938	-179
147	-179	403	-177	659	-178	915	-178	1171	-178	1427	-180	1683	-178	1939	-177
148	-182	404	-178	660	-179	916	-182	1172	-179	1428	-178	1684	-183	1940	-178
149	-178	405	-179	661	-177	917	-177	1173	-180	1429	-178	1685	-178	1941	-179
150	-179	406	-177	662	-180	918	-177	1174	-178	1430	-178	1686	-178	1942	-178
151	-179	407	-177	663	-178	919	-177	1175	-178	1431	-179	1687	-178	1943	-177
152	-178	408	-178	664	-179	920	-178	1176	-178	1432	-179	1688	-178	1944	-178
153	-179	409	-182	665	-182	921	-179	1177	-183	1433	-178	1689	-182	1945	-178
154	-177	410	-178	666	-180	922	-181	1178	-178	1434	-177	1690	-181	1946	-180
155	-178	411	-178	667	-180	923	-178	1179	-177	1435	-183	1691	-177	1947	-179
156	-179	412	-177	668	-177	924	-181	1180	-177	1436	-177	1692	-178	1948	-183
157	-180	413	-178	669	-178	925	-178	1181	-178	1437	-177	1693	-180	1949	-178
158	-179	414	-178	670	-177	926	-177	1182	-183	1438	-177	1694	-179	1950	-177
159	-180	415	-182	671	-181	927	-178	1183	-177	1439	-178	1695	-178	1951	-178
160	-179	416	-180	672	-179	928	-179	1184	-177	1440	-179	1696	-181	1952	-178
161	-179	417	-178	673	-177	929	-179	1185	-180	1441	-180	1697	-178	1953	-179

Table 8 continuation from previous page.

Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.
162	-178	418	-178	674	-178	930	-178	1186	-180	1442	-181	1698	-177	1954	-178
163	-178	419	-182	675	-180	931	-182	1187	-178	1443	-179	1699	-178	1955	-182
164	-181	420	-179	676	-178	932	-179	1188	-178	1444	-177	1700	-177	1956	-178
165	-178	421	-178	677	-179	933	-178	1189	-178	1445	-180	1701	-178	1957	-178
166	-178	422	-178	678	-179	934	-182	1190	-177	1446	-178	1702	-179	1958	-179
167	-178	423	-182	679	-177	935	-178	1191	-178	1447	-177	1703	-179	1959	-178
168	-178	424	-180	680	-180	936	-181	1192	-183	1448	-177	1704	-177	1960	-178
169	-180	425	-178	681	-178	937	-179	1193	-180	1449	-179	1705	-178	1961	-179
170	-178	426	-179	682	-179	938	-178	1194	-177	1450	-178	1706	-177	1962	-179
171	-179	427	-179	683	-178	939	-178	1195	-180	1451	-178	1707	-178	1963	-183
172	-178	428	-178	684	-178	940	-179	1196	-180	1452	-179	1708	-179	1964	-178
173	-178	429	-177	685	-179	941	-177	1197	-178	1453	-180	1709	-177	1965	-178
174	-178	430	-181	686	-177	942	-177	1198	-179	1454	-178	1710	-178	1966	-182
175	-178	431	-179	687	-179	943	-183	1199	-178	1455	-179	1711	-177	1967	-179
176	-180	432	-180	688	-178	944	-179	1200	-179	1456	-180	1712	-180	1968	-179
177	-178	433	-178	689	-178	945	-179	1201	-180	1457	-180	1713	-183	1969	-178
178	-177	434	-181	690	-178	946	-181	1202	-178	1458	-180	1714	-181	1970	-178
179	-179	435	-183	691	-178	947	-178	1203	-178	1459	-182	1715	-178	1971	-178
180	-178	436	-178	692	-178	948	-183	1204	-182	1460	-177	1716	-178	1972	-178
181	-177	437	-181	693	-177	949	-178	1205	-178	1461	-178	1717	-178	1973	-182
182	-183	438	-178	694	-181	950	-178	1206	-178	1462	-177	1718	-177	1974	-178
183	-179	439	-182	695	-182	951	-177	1207	-177	1463	-179	1719	-178	1975	-180
184	-178	440	-178	696	-179	952	-179	1208	-178	1464	-179	1720	-178	1976	-178
185	-182	441	-180	697	-178	953	-178	1209	-178	1465	-179	1721	-178	1977	-179
186	-182	442	-179	698	-181	954	-178	1210	-177	1466	-178	1722	-179	1978	-178
187	-178	443	-179	699	-181	955	-178	1211	-181	1467	-178	1723	-178	1979	-178
188	-181	444	-180	700	-181	956	-178	1212	-177	1468	-178	1724	-177	1980	-177
189	-178	445	-178	701	-178	957	-180	1213	-177	1469	-179	1725	-178	1981	-178
190	-179	446	-179	702	-179	958	-182	1214	-177	1470	-178	1726	-181	1982	-179
191	-179	447	-178	703	-180	959	-179	1215	-179	1471	-177	1727	-178	1983	-178
192	-177	448	-179	704	-178	960	-182	1216	-178	1472	-177	1728	-180	1984	-177
193	-178	449	-177	705	-179	961	-178	1217	-178	1473	-177	1729	-178	1985	-177
194	-178	450	-177	706	-180	962	-178	1218	-181	1474	-181	1730	-178	1986	-178
195	-178	451	-179	707	-177	963	-178	1219	-178	1475	-178	1731	-178	1987	-178
196	-177	452	-178	708	-178	964	-182	1220	-179	1476	-177	1732	-178	1988	-178
197	-177	453	-178	709	-178	965	-179	1221	-180	1477	-178	1733	-177	1989	-178
198	-180	454	-179	710	-178	966	-179	1222	-178	1478	-178	1734	-179	1990	-179
199	-178	455	-178	711	-178	967	-178	1223	-179	1479	-179	1735	-178	1991	-178
200	-180	456	-180	712	-178	968	-178	1224	-177	1480	-177	1736	-179	1992	-178

Table 8 continuation from previous page.

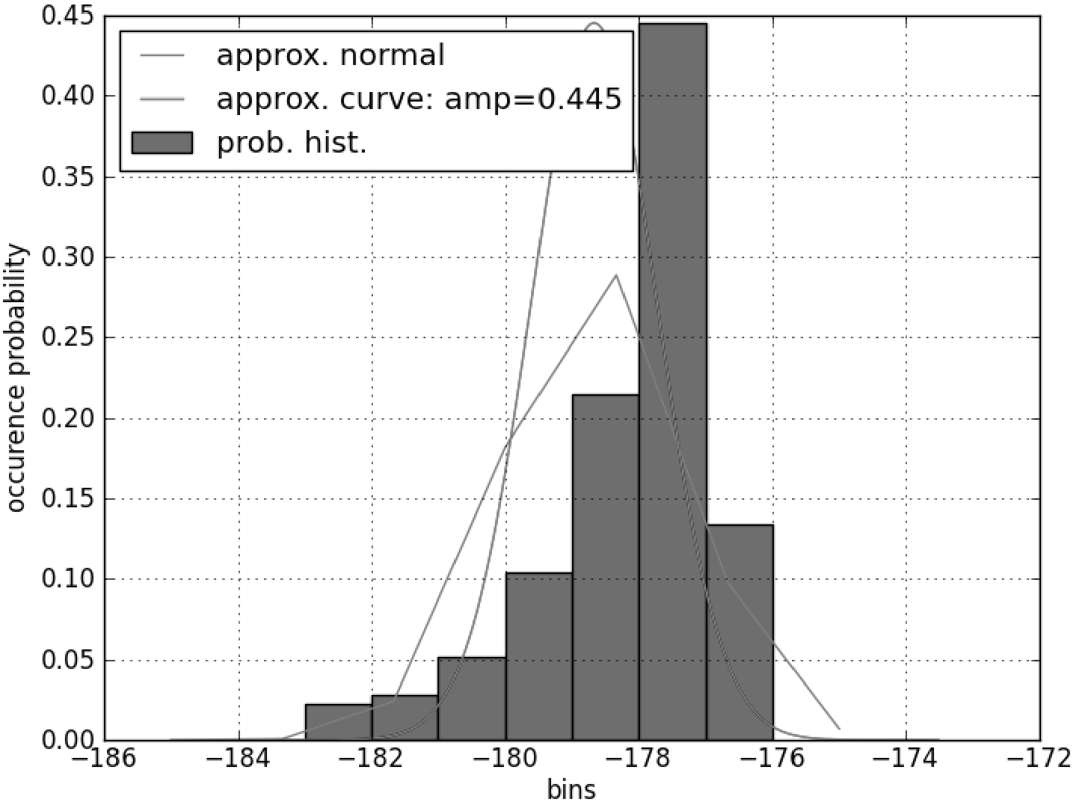
Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.
201	-179	457	-179	713	-177	969	-178	1225	-177	1481	-178	1737	-178	1993	-178
202	-181	458	-179	714	-178	970	-178	1226	-178	1482	-178	1738	-178	1994	-178
203	-178	459	-178	715	-179	971	-177	1227	-177	1483	-178	1739	-179	1995	-178
204	-178	460	-179	716	-179	972	-178	1228	-178	1484	-177	1740	-179	1996	-177
205	-180	461	-179	717	-183	973	-180	1229	-181	1485	-179	1741	-178	1997	-179
206	-179	462	-178	718	-178	974	-178	1230	-180	1486	-179	1742	-179	1998	-179
207	-181	463	-178	719	-178	975	-179	1231	-178	1487	-178	1743	-178	1999	-177
208	-180	464	-177	720	-177	976	-181	1232	-179	1488	-178	1744	-178	2000	-178
209	-179	465	-182	721	-178	977	-179	1233	-178	1489	-179	1745	-177	2001	-178
210	-180	466	-181	722	-178	978	-177	1234	-178	1490	-180	1746	-178	2002	-181
211	-179	467	-178	723	-179	979	-178	1235	-178	1491	-178	1747	-179	2003	-178
212	-178	468	-177	724	-177	980	-178	1236	-178	1492	-178	1748	-178	2004	-178
213	-179	469	-178	725	-179	981	-179	1237	-177	1493	-178	1749	-179	2005	-178
214	-178	470	-178	726	-178	982	-178	1238	-178	1494	-182	1750	-179	2006	-178
215	-180	471	-180	727	-178	983	-178	1239	-179	1495	-180	1751	-179	2007	-178
216	-183	472	-179	728	-178	984	-178	1240	-179	1496	-179	1752	-180	2008	-178
217	-178	473	-179	729	-178	985	-178	1241	-180	1497	-178	1753	-178	2009	-178
218	-178	474	-178	730	-177	986	-179	1242	-178	1498	-180	1754	-179	2010	-179
219	-179	475	-177	731	-177	987	-177	1243	-179	1499	-178	1755	-177	2011	-178
220	-178	476	-177	732	-180	988	-178	1244	-178	1500	-181	1756	-180	2012	-178
221	-177	477	-177	733	-180	989	-180	1245	-178	1501	-179	1757	-179	2013	-178
222	-178	478	-177	734	-181	990	-177	1246	-179	1502	-183	1758	-178	2014	-178
223	-179	479	-179	735	-179	991	-178	1247	-177	1503	-178	1759	-178	2015	-178
224	-178	480	-178	736	-178	992	-178	1248	-178	1504	-178	1760	-179	2016	-178
225	-177	481	-178	737	-178	993	-178	1249	-177	1505	-177	1761	-177	2017	-177
226	-180	482	-181	738	-178	994	-178	1250	-179	1506	-179	1762	-178	2018	-177
227	-178	483	-178	739	-179	995	-177	1251	-178	1507	-177	1763	-179	2019	-178
228	-178	484	-177	740	-178	996	-178	1252	-177	1508	-178	1764	-178	2020	-183
229	-178	485	-179	741	-178	997	-181	1253	-177	1509	-179	1765	-178	2021	-181
230	-179	486	-179	742	-179	998	-178	1254	-178	1510	-178	1766	-179	2022	-177
231	-178	487	-177	743	-177	999	-179	1255	-177	1511	-178	1767	-180	2023	-178
232	-180	488	-177	744	-177	1000	-179	1256	-177	1512	-177	1768	-178	2024	-178
233	-178	489	-178	745	-178	1001	-178	1257	-178	1513	-177	1769	-180	2025	-178
234	-179	490	-178	746	-179	1002	-179	1258	-179	1514	-178	1770	-180	2026	-178
235	-179	491	-178	747	-179	1003	-178	1259	-178	1515	-178	1771	-179	2027	-178
236	-178	492	-179	748	-179	1004	-178	1260	-179	1516	-178	1772	-179	2028	-180
237	-181	493	-178	749	-178	1005	-178	1261	-180	1517	-178	1773	-180	2029	-181
238	-181	494	-177	750	-177	1006	-178	1262	-178	1518	-178	1774	-179	2030	-178
239	-178	495	-177	751	-178	1007	-178	1263	-178	1519	-177	1775	-181	2031	-179

Table 8 continuation from previous page.

Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.	Obs.	Val.
240	-178	496	-177	752	-183	1008	-178	1264	-178	1520	-179	1776	-177	2032	-180
241	-180	497	-177	753	-180	1009	-177	1265	-179	1521	-178	1777	-179	2033	-178
242	-178	498	-178	754	-179	1010	-181	1266	-180	1522	-180	1778	-177	2034	-182
243	-177	499	-178	755	-178	1011	-178	1267	-181	1523	-181	1779	-178	2035	-179
244	-179	500	-181	756	-178	1012	-180	1268	-180	1524	-179	1780	-178	2036	-178
245	-180	501	-179	757	-179	1013	-177	1269	-177	1525	-178	1781	-178	2037	-179
246	-178	502	-179	758	-178	1014	-178	1270	-178	1526	-178	1782	-177	2038	-178
247	-178	503	-178	759	-178	1015	-178	1271	-183	1527	-179	1783	-178	2039	-178
248	-179	504	-178	760	-179	1016	-181	1272	-179	1528	-178	1784	-182	2040	-178
249	-179	505	-178	761	-178	1017	-179	1273	-178	1529	-178	1785	-179	2041	-178
250	-180	506	-182	762	-178	1018	-179	1274	-180	1530	-178	1786	-178	2042	-178
251	-179	507	-183	763	-178	1019	-178	1275	-178	1531	-179	1787	-178	2043	-182
252	-178	508	-179	764	-180	1020	-178	1276	-179	1532	-178	1788	-178	2044	-177
253	-182	509	-180	765	-178	1021	-177	1277	-179	1533	-179	1789	-178	2045	-178
254	-182	510	-178	766	-179	1022	-180	1278	-178	1534	-178	1790	-179	2046	-180
255	-177	511	-178	767	-178	1023	-180	1279	-179	1535	-177	1791	-179	2047	-178

Source: author (2021).

Figure 15: Histogram of observations, and Gaussian approximation.



Source: author (2021).

A.3 The cumulative distribution table used

This appendix section presents in Table 9 the cumulative distribution table (CDT) used in the tests of the CDT-based discrete Gaussian sampler.

Table 9: The cumulative distribution table used by this work.

Position	Value	Position	Value
0	0	97	340282366902788985317896529203028560353
1	9228462493351110473597041657947797092	98	340282366909461681135086961316997321092
2	27642789659143819934692025361980055762	99	340282366913713946414412559132152934149
3	45929912339185481206974167095818864061	100	340282366916411265894161291135121729086
4	64006977247976486123651496194149979097	101	340282366918114355282467857200614032189
5	81794008662808119598967550596620492198	102	340282366919184729187249640618211905803

Table 9 continuation from previous page.

Position	Value	Position	Value
6	99214967010543519719527723982122616624	103	340282366919854346383310497515313483120
7	116198718570185839248173721556362105814	104	340282366920271322037738061722552979678
8	132679895809032129485237851807443093831	105	340282366920529778750018355527593467690
9	148599631277039782639814771967288384067	106	340282366920689241022080314156838418824
10	163906151856162374745290895274741469911	107	340282366920787172226455454882666338026
11	178555224346767839709539400552399419832	108	340282366920847037815352332287494207629
12	192510447711147483407448520742454256647	109	340282366920883465069693266007857866034
13	205743391622772455628744117035017830165	110	340282366920905528276344232158095493526
14	218233585133794971304219977848862993467	111	340282366920918829875096654603813664759
15	229968363129746213565316406163284007235	112	340282366920926812249040363447839731974
16	240942581665693854082205362525868016057	113	340282366920931580436382437767102927622
17	251158216172002921770481550717083362665	114	340282366920934415531066409418642660678
18	260623858806746505114691852226601027586	115	340282366920936093464935720427947186385
19	269354132870709165905526117913527246739	116	340282366920937081961419531051917797029
20	277369043173590454192770916761457234520	117	340282366920937661614879909918756760010
21	284693281557884949750179305992168359467	118	340282366920937999955976427751912035217
22	291355506486676852442751272614702640149	119	340282366920938196533603231539009314689
23	297387614741713670208101001790230752188	120	340282366920938310219459865522637422106
24	302824021934610738559622041384292999210	121	340282366920938375663756591334491989430
25	307700966795635144557768932918957406644	122	340282366920938413163663289681020643343
26	312055852167739877852802764089557951894	123	340282366920938434552225187682149463164
27	315926633397728357937842503713030363383	124	340282366920938446695225278270824484201
28	319351262479228085846041229665079340378	125	340282366920938453557426158857674219280
29	322367193955305743664346087034485275350	126	340282366920938457417484527860251694780
30	325010956314605435946813411768504745518	127	340282366920938459578795867256495467918
31	327317790484558076752574972980945282764	128	340282366920938460783370762814764232092
32	329321355095645417755941825328303916953	129	340282366920938461451627428274575360296
33	331053496504581822367938598164939798159	130	340282366920938461820643946068958880120
34	332544080149682245136832795231139185736	131	340282366920938462023478214034466375672
35	333820878682692463596365006169060202807	132	340282366920938462134454418509984731758
36	334909511479213505440293948529744414348	133	340282366920938462194892604558729646256
37	335833429564599848145021829589587742578	134	340282366920938462227655780254438251843
38	336613939684627828788307393185703765487	135	340282366920938462245334612545605103389
39	337270261173934837195825998906436035728	136	340282366920938462254830032492351639936
40	337819609398823282862144715854695818333	137	340282366920938462259906572265589767112
41	338277299840159867617134524799962596041	138	340282366920938462262608131069520674070
42	338656867301382879444154781925233325266	139	340282366920938462264039178646791852691
43	338970195241265506721993563170537548183	140	340282366920938462264793726409988784362
44	339227650808253032038440903731776363832	141	340282366920938462265189742120070265265

Table 9 continuation from previous page.

Position	Value	Position	Value
45	339438221763109584782516034148317213384	142	340282366920938462265396628123152951141
46	339609652093265732827765441962657935273	143	340282366920938462265504210917267147280
47	339748573723863218674739391754985659825	144	340282366920938462265559897112441174990
48	339860632299656076346434833343566197087	145	340282366920938462265588588086088029936
49	339950605535596306043330422086896686103	146	340282366920938462265603302265377355723
50	340022513103149665187694553655575855342	147	340282366920938462265610813646807090556
51	340079717428902553538171454761371537605	148	340282366920938462265614630422206142293
52	340125015129807829022926769953448126673	149	340282366920938462265616560907102454639
53	340160719096135231850585043070367197286	150	340282366920938462265617532824212264950
54	340188731461631912016860203677173672351	151	340282366920938462265618019887126183316
55	340210607874941583169578684213845612839	152	340282366920938462265618262846641953090
56	340227613612444904872291704014997276637	153	340282366920938462265618383482322801688
57	340240772156474223096304505698880551789	154	340282366920938462265618443104888604256
58	340250906910681973641524361347082489522	155	340282366920938462265618472436678069855
59	340258676742503432739092215018291397605	156	340282366920938462265618486800151397578
60	340264606037148044339979539544772440333	157	340282366920938462265618493801366179249
61	340269109923897641173573541977057476264	158	340282366920938462265618497198247277555
62	340272515298608963745510402213100540331	159	340282366920938462265618498838762654389
63	340275078220479752248548421531777799685	160	340282366920938462265618499627392631141
64	340276998209934281211602988011139621644	161	340282366920938462265618500004755593342
65	340278429920846833669409456228479717399	162	340282366920938462265618500184492920557
66	340279492606561684163118977281270278900	163	340282366920938462265618500269706792993
67	340280277747035410469601142564771442911	164	340282366920938462265618500309920613934
68	340280855155178466467342323362955809018	165	340282366920938462265618500328810676378
69	340281277834944774093105230899685556540	166	340282366920938462265618500337643192989
70	340281585822400801920940256530947823350	167	340282366920938462265618500341754013813
71	340281809204112302163003653234724474176	168	340282366920938462265618500343658446594
72	340281970474717317553901953463590866790	169	340282366920938462265618500344536651309
73	340282086367350892940184522243560420621	170	340282366920938462265618500344939756969
74	340282169266383848123707234149974020989	171	340282366920938462265618500345123933820
75	340282228291399701676933390984801831882	172	340282366920938462265618500345207695273
76	340282270124088138432650281313180044007	173	340282366920938462265618500345245613357
77	340282299635395903851266206075114275335	174	340282366920938462265618500345262699403
78	340282320358468582862967655190722194527	175	340282366920938462265618500345270362949
79	340282334843280509558140703121923758314	176	340282366920938462265618500345273784405
80	340282344921053857622417845235828264621	177	340282366920938462265618500345275304900
81	340282351900313069332539850050451495281	178	340282366920938462265618500345275977493
82	340282356711443000621385556860036325483	179	340282366920938462265618500345276273643
83	340282360012688778320785941847082130970	180	340282366920938462265618500345276403440

Table 9 continuation from previous page.

Position	Value	Position	Value
84	340282362267455660587782539040615659220	181	340282366920938462265618500345276460065
85	340282363800371993390626989531164892809	182	340282366920938462265618500345276484654
86	340282364837729063796944129682366535053	183	340282366920938462265618500345276495282
87	340282365536494004322210194491509825597	184	340282366920938462265618500345276499855
88	340282366005012734817613777873088096491	185	340282366920938462265618500345276501813
89	340282366317704053877358501294218166755	186	340282366920938462265618500345276502647
90	340282366525433329569361423227664819231	187	340282366920938462265618500345276503001
91	340282366662797227047845184570220926209	188	340282366920938462265618500345276503150
92	340282366753212233632867345370957481661	189	340282366920938462265618500345276503212
93	340282366812450378220363228485688267561	190	340282366920938462265618500345276503238
94	340282366851083107627671162948427431413	191	340282366920938462265618500345276503248
95	340282366876161653578565443519559930409	192	340282366920938462265618500345276503252
96	340282366892366403327902636518806646673	193	340282366920938462265618500345276503253

Source: author (2021).

A.4 Cumulative distribution table results

Table 10 presents all of the discrete Gaussian sampling values, regarding the cumulative distribution table sampling strategy.

Table 10: Sampling results for the cumulative distribution table implementation.

β	s	**elapsed time (ms)**	**CPU cycles ×10^6**	$\mu_{\mathcal{D}}$	$\sigma_{\mathcal{D}}$	**skewness**	**kurtosis**	**relative entropy**
1	2	7.8	16.40	21.0	15.0	0	-2	∞
2	4	7.8	16.42	-1.5	4.5	0	-2	∞
3	8	7.8	16.40	9.0	19.0	0	-2	∞
4	16	6.8	14.35	7.0	12.0	0	-2	0.3465735903
5	32	6.8	14.35	4.0	17.0	0	-2	0.4158883083
6	64	6.3	13.32	11.0	10.0	0	-2	0.4852030264
7	128	6.1	12.81	-9.0	17.0	0	-2	0.3465735903
8	256	6.1	12.82	-8.0	10.0	0	-2	0.4628886713
9	512	6.6	13.95	-1.0	12.0	0	-2	0.3564175976
10	1024	6.5	13.72	-13.5	2.5	0	-2	0.3732798688
11	2048	6.4	13.49	-13.0	8.0	0	-2	0.3608235965
12	4096	6.4	13.41	-4.5	10.5	0	-2	0.3538642674
13	8192	6.4	13.60	-1.0	10.0	0	-2	0.3663805817
14	16384	6.3	13.31	-22.0	1.0	0	-2	0.3603741576
15	32768	6.3	13.33	4.0	26.0	0	-2	0.3557731270
16	65536	6.8	14.35	-2.0	2.0	0	-2	0.3590101838
17	131072	6.7	14.13	-3.5	23.5	0	-2	0.3630296207
18	262144	6.8	14.34	-12.5	7.5	0	-2	0.3596341891
19	524288	6.7	14.20	3.5	2.5	0	-2	0.3599337052
20	1048576	6.7	14.19	-17.0	15.0	0	-2	0.3609803212

Source: author (2021).

A.5 Fast Walsh–Hadamard transform results

Table 11 presents all of the discrete Gaussian sampling values, regarding the fast Walsh–Hadamard transform sampling strategy.

Table 11: Sampling results for the fast Walsh–Hadamard transform implementation.

β	s	**elapsed time (μs)**	**CPU cycles ×10^3**	$\mu_{\mathfrak{D}}$	$\sigma_{\mathfrak{D}}$	**skewness**	**kurtosis**	**relative entropy**
0	1	00.0	0.00	0	02	0	-2	∞
1	2	00.0	0.00	0	03	0	-2	∞
2	4	00.0	0.00	0	02	0	-2	∞
3	8	00.0	0.00	0	12	0	-2	∞
4	16	00.0	0.00	0	18	0	-2	0.2772588722
5	32	00.0	0.00	0	17	0	-2	0.2079441542
6	64	00.0	0.00	0	11	0	-2	0.3060270795
7	128	00.0	0.00	0	10	0	-2	0.3465735903
8	256	61.0	128.10	0	02	0	-2	0.3465735903
9	512	30.5	64.050	0	12	0	-2	0.3060270795
10	1024	45.8	96.18	0	02	0	-2	0.3019448800
11	2048	45.8	96.18	0	02	0	-2	0.3334159545
12	4096	49.6	104.16	0	14	0	-2	0.3320553893
13	8192	52.7	110.67	0	12	0	-2	0.3252011573
14	16384	59.1	124.11	0	10	0	-2	0.3164474572
15	32768	54.4	114.24	0	10	0	-2	0.3206531674
16	65536	53.2	111.72	0	06	0	-2	0.3195664039
17	131072	53.1	111.51	0	15	0	-2	0.3202583187
18	262144	52.3	109.83	0	08	0	-2	0.3176651006
19	524288	53.2	111.72	0	00	0	-3	0.3185097623
20	1048576	52.0	109.20	0	18	0	-2	0.3176811012
21	2097152	51.7	108.57	0	07	0	-2	0.3179489186
22	4194304	51.3	107.73	0	02	0	-2	0.3184993903
23	8388608	51.6	108.36	0	12	0	-2	0.3181674804

Source: author (2021).