Kernel Density Estimation on Homomorphically Encrypted Data

Máté Kormos

MPhil Thesis, August 2020
Tinbergen Institute, The Netherlands

Supervisor:
Charles S. Bos

Committee:
Charles S. Bos
Paolo Gorgi
Florian Wagener[¶]

[¶]Vrije Universiteit Amsterdam, Vrije Universiteit Amsterdam, Universiteit van Amsterdam, respectively.

Abstract

I propose a method for univariate Kernel Density Estimation (KDE), for bounded support probability density functions, on Homomorphically Encrypted (HE) data. The estimator, called HE-KDE, facilitates outsourcing nonparametric density estimation to a semi-honest cloud without exposing the non-encrypted data. HE-KDE locally approximates a kernel function with a polynomial. In contrast to kernels typically used for non-encrypted KDE, the polynomial can be natively evaluated on ring-homomorphically encrypted data.

The density estimator is non-negative, and its integral is bounded by one. However, it is defective as it need not integrate to one. There is a trade-off between minimising the Mean Integrated Square Error (MISE) of the estimator and maximising encryption security. A polynomial balancing these two goals is designed. Asymptotically, it ensures that the MISE converges to zero, and provides encryption security.

Applications on simulated data are presented, using Homomorphic Encryption for Arithmetic of Approximate Numbers (HEAAN, Cheon et al. (2017)).

Acknowledgements

I am grateful to my supervisor, Charles Bos, for the (video) discussions we had and for his helpful comments shaping this thesis. I thank Yongsoo Song of Cheon et al. (2017) for clarifications about the CKKS scheme. The thesis largely draws on the much-appreciated work of people contributing to open science and open source coding. Above all, I am indebted to my family for their support.

Contents

List of Tables **v**

List of Figures **vi**

List of Propositions **vii**

List of Algorithms **viii**

1 Introduction **1**

1.1 General Notation 2

2 Homomorphic Encryption **2**

2.1 CKKS Scheme (HEAAN) 3

2.1.1 Overview 3

2.1.2 Notation 4

2.1.3 Complete Scheme 6

2.2 Applications in Statistics 9

3 Kernel Density Estimation **11**

3.1 Description 11

3.2 HE Difficulties 11

3.3 HE-KDE 12

3.4 HE-KDE Properties 15

3.5 Choosing Parameters: Statistical versus Encryption Optimality 17

3.5.1 Statistical Optimality 17

3.5.2 Encryption Optimality 19

4 Examples **21**

4.1 Non-encrypted Data 21

4.2 Encrypted Data 25

4.2.1 Is the Bandwidth Too Conservative? 28

5 Conclusion **29**

References **30**

Appendix A Proofs **33**

A.1 Proposition 1 33

A.2 Proposition 2 33

A.3 Proposition 3 35
A.4 Proposition 4 37

Appendix B Algorithms **38**

Appendix C Figures **39**
C.1 Parameter Choices 39
C.1.1 Statistical Optimality 39
C.1.2 Encryption Optimality 41
C.2 HE-KDE Examples 43
C.2.1 Non-encrypted Data 43
C.2.2 Encrypted Data 46

List of Tables

Table 1 Cumulative Number of HE Multiplications in Algorithm 2 33

List of Figures

Figure 1 Homomorphic Encryption Workflow 1
Figure 2 Operations with Homomorphic Encryption 9
Figure 3 Local Approximate Kernel 13
Figure 4 Effect of Parameters on Local Approximate Kernel Properties 13
Figure 5 Statistically Optimal Parameter Choices; $S = [0, 1.5]$, $\beta_f = 3$. 19
Figure 6 Encryption-optimal Parameter Choices; $S = [0, 100]$, $\beta_f = 3$. 20
Figure 7 HE-KDE Non-encrypted Examples 23
Figure 8 HE-KDE Encrypted Examples 26
Figure 9 Empirical Bias of HE-KDE as a Function of Bandwidth 29
Figure 10 Local Approximate Kernel: Sign Properties 34
Figure 11 Integrals I_1, I_2 of Proposition 3 39
Figure 12 Statistically Optimal Parameter Choices for Different S 40
Figure 13 Encryption-optimal Parameter Choices for Different S 41
Figure 14 HE-KDE More Non-encrypted Examples 43
Figure 15 HE-KDE More Encrypted Examples 46

List of Propositions

Proposition 1 **Multiplicative Depth of Algorithm 2** 15
Proposition 2 **Bounds** 16
Proposition 3 **MISE** 16
Proposition 4 **Optimal Bandwidth** 17

List of Algorithms

Algorithm 1 **HE-KDE (CKKS) Step 1: Client encrypts data** 14
Algorithm 2 **HE-KDE (CKKS) Step 2: Cloud computes estimator** . . . 15
Algorithm 3 **HE-KDE (CKKS) Step 3: Client decrypts estimator** . . . 15
Algorithm 4 **Sum** 38

1. Introduction

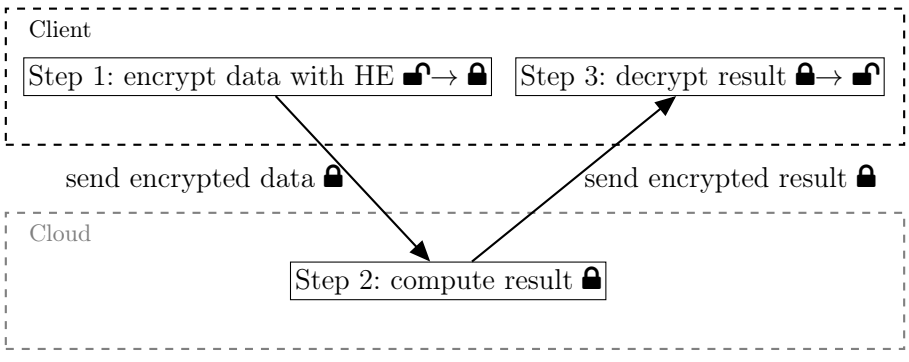
Consider a client, who has sensitive data, and a cloud provider, who has large computational capacity. The client wants to outsource statistical computation to the cloud while protecting the data. Homomorphic Encryption (HE) facilitates computation on encrypted data, providing data protection as follows. The client encrypts the data with HE. The encrypted data are sent to the cloud, which does the computation. The result, still encrypted, is sent back to the client, who decrypts it (see Figure 1). During this process only the client sees the non-encrypted data, so the data are protected.

HE has been applied with a number of statistical methods (Yang et al., 2019), but applications with nonparametric statistics are rare. In my thesis, the use of HE with Kernel Density Estimation (KDE), a popular nonparametric method for data exploration, is addressed. To the best of my knowledge, this work is the first to devise kernel density estimation on homomorphically encrypted data.

I propose a method for univariate density estimation where the support of the estimated density is a bounded real interval. The estimator, called HE-KDE, is based on a polynomial that can be *natively* evaluated on homomorphically encrypted data more efficiently than kernels used on non-encrypted data. The bias, and upper bounds on the variance and the Mean Integrated Square Error (MISE) are derived analytically together with the MISE-optimal bandwidth. There is a trade-off between minimising MISE and maximising security of the encryption with respect to the polynomial. A polynomial balancing these two goals is designed, ensuring MISE convergence and encryption security when the sample size used for estimation goes to infinity.

General notation is introduced in the next subsection. Section 2 describes HE and its applications. Section 3 describes why and how kernel density estimation is made compatible with HE. Section 4 contains applications of HE-KDE to simulated data.[1]

Figure 1: Homomorphic Encryption Workflow



Notes: 🔒 denotes encrypted, 🔓 denotes non-encrypted data.

[1]Codes are available at `github.com/kmmate/mphil_thesis_codes`

1.1. General Notation

Symbols $\implies$, $\impliedby$ mean *implies* and implied by, respectively. $\iff$ means *if and only if*. Symbols $\forall$ and $\exists$ mean *for all* and *there exists*, respectively. Symbols $=$ and $\neq$ mean *equal* and *not equal*, respectively. $a \coloneqq b$ means set a equal to b, or set a equal to the output of b if b is an algorithm. $b =: a$ means the same thing. $a \to b$ either specifies a function mapping from a to b or the convergence of a to b. $\{a, b\}$ is a set with elements a, b. $|a|$ is the cardinality of set a. $a \in b$ ($a \notin b$) means that a is (not an) element of a set b. For sets a, b, $a \cap b$ is the intersection, $a \cup b$ is the union; $a \subset b$ means that a is a subset of b.

$-a$ and a^{-1} are the additive and multiplicative inverses of a, respectively. 0 and 1 are the additive and multiplicative idenities, respectively. $a + b$, $a - b$, $a \cdot b$, a/b are addition, substraction (addition to additive inverse of b), multiplication and division (multiplication with multiplicative inverse of b), respectively. $\mathbb{1}$ is the indicator function. lim, $\sum$, $\int$, mod, $\log_a$ are the limit, summation, integral, modulo and logarithm of base a operations, respectively. $\gcd(a, b)$ is the greatest common divisor of scalars a, b. inf means infimum, sup means supremum. min is minimum, max is maximum. ∞ is infinity.

$\mathbb{C}$, $\mathbb{N}$ and $\mathbb{R}$ denote the complex, natural and real numbers, respectively. $0 \in \mathbb{N}$. Let $\mathbb{N}_{\geq a}, \mathbb{R}_{\geq a}$ denote all naturals and reals, respectively, larger than or equal to $a \in \mathbb{R}$. Let this definition naturally extend to $\leq, <, >$. $\mathbb{R}_+, \mathbb{R}_{++}$ stand for $\mathbb{R}_{\geq 0}, \mathbb{R}_{>0}$, respectively. For $a \in \mathbb{C}$, $|a|$ is the absolute value of a, $\bar{a}$ is the complex conjugate. $[a, b]$ with $a, b \in \mathbb{R}, a \leq b$ is a closed interval. For $a \in \mathbb{R}_+$, $[\pm a] \coloneqq [-a, a]$. All vectors are column vectors, and are denoted by $\boldsymbol{a} = (a_j)_{j \in \mathcal{I}} \in b^{|\mathcal{I}|}$ with elements $a_i \in b$, $i \in \mathcal{I}$, for some ordered index set $\mathcal{I}$ and some set b. For $n \in \mathbb{N}_{\geq 1}$, $[n] \coloneqq \{1, 2, \ldots, n\}$ is an ordered (index) set. For $n \in \mathbb{N}_{\geq 1}$, $\mathbf{1}_n \in \mathbb{R}^n$ is an n-long vector of ones. For a vector $\boldsymbol{a} = (a_j)_{j \in \mathcal{I}}$ and $k \in \mathbb{R}$, $\boldsymbol{a}^k \coloneqq (a_j^k)_{j \in \mathcal{I}}$ means element-wise exponentiation. For scalar b and vector $\boldsymbol{a} = (a_j)_{j \in \mathcal{I}}$, $b \cdot \boldsymbol{a} = (ba_j)_{j \in \mathcal{I}}$ is element-wise multiplication. The dot $\cdot$ is sometimes suppressed.

For functions $f : b \to c$ and $g : a \to b$, $(f \circ g) : a \to c$ is a function composition. For functions $f, g : \mathbb{R} \to \mathbb{R}$, $x \in \mathbb{R}$, $f(x)$ is proportional to $g(x)$, denoted by $f(x) \propto g(x)$, if there exists $a \in \mathbb{R}_{++}$ such that $f(x) = ag(x)$. For $x \in \mathbb{R}_+$, $f : \mathbb{R}_+ \to \mathbb{R}_+$ and $g : \mathbb{R}_+ \to \mathbb{R}_{++}$, $f(x) = O(g(x))$ means that there exist $a, x_0 \in \mathbb{R}_+$ such that $f(x) \leq ag(x)$ for all $x \geq x_0$. $g(x) = \Omega(f(x))$ means the same thing. $f(x) = o(g(x))$ means that $f(x)/g(x) \to 0$ as x approaches some limit (typically 0 or ∞). $\mathbb{E}[.]$ ($\mathbb{E}_f[.]$) is expectation, $\mathbb{V}[.]$ ($\mathbb{V}_f[.]$) is variance (with respect to density f).

2. Homomorphic Encryption

Gentry (2009) was the first to design a (fully) homomorphic encryption scheme (Homomorphic-Encryption.org). Since then a few HE schemes were proposed, including but not limited to: BGV (Brakerski et al., 2011), BFV (Fan and Vercauteren, 2012), an NTRU-based (Lopez-Alt et al., 2013), GSW Gentry et al. (2013), and the CKKS (also called homomorphic encryption for

arithmetic of approximate numbers, HEAAN, Cheon et al. (2017)) schemes. See Albrecht et al. (2018) for an overview. I use the CKKS scheme. Even though it gives only approximate results, I choose it because (i) it performs well in statistical applications in terms of computation time and precision (see Kim et al. (2018b), Kim et al. (2018a) for logit, and Jiang et al. (2018) for neural networks applications); and (ii) it provides an efficient way of encoding a vector by the batching technique (Cheon et al. (2017), see below). In the rest of this section, I describe the CKKS scheme.

2.1. CKKS Scheme (HEAAN)

In this subsection, a non-technical overview of how CKKS works is given, followed by a formal explanation.

2.1.1 Overview

Suppose we have the same application described in the Introduction (Figure 1). Specifically, we have a vector of complex or real numbers we want to encrypt and send to the cloud to perform computation on the encrypted vector. Then CKKS involves the following steps.

First, in what can be thought of as a preprocessing step, the vector is encoded. *Encoding* maps the complex vector to a polynomial in a polynomial ring. The inverse of encoding is decoding, which maps a polynomial into a complex vector. Encoding and decoding are homomorphisms between the complex vector space and the polynomial ring – hence the name of HE. It is because of the homomorphism that computations can be performed on encrypted data: adding (multiplying) two polynomials then decoding the result leads to the same result as adding (multiplying) two complex vectors element-wise. Importantly, note that while addition/multiplication is enabled by HE, other operations, for e.g. multiplicative inverse (i.e. division) or exponential function are *not* natively supported.

Second, the encoded vector, now a polynomial, is encrypted to what is called *ciphertext*, which is a vector of polynomials. *Encryption* can be done with the *public key*, which is public information. The inverse operation of encryption is decryption. Decryption is only possible with the *secret key*, which is private information only accessible to the data owner. Therefore, encryption provides data protection as long as the secret key is indeed secret and secure enough to withstand attacks. CKKS uses polynomial rings because then successful attacks to break the encryption need to solve the Ring Learning With Errors (RLWE) problem. RLWE is assumed to be a hard problem, withstanding even "powerful attacks". (Cheon et al., 2017).

Third, the encrypted polynomial is sent to the cloud, where the computation happens. *Homomorphic computations* usually require the *evaluation key*, and *switching key*, which are public information. Last, the cloud sends back the encrypted result, which is first decrypted with the secret key, and then decoded. The decoded result is a vector of complex numbers.

2.1.2 Notation

Rings, encoding, and probability distributions have a central role in CKKS. In the following, formal descriptions of these based on Cheon et al. (2017) are given, and some examples are provided.

Rings For $q \in \mathbb{Z}$, let $\mathbb{Z}_q \coloneqq \mathbb{Z}/q\mathbb{Z}$ be the ring of integers modulo q. The congruence class representatives are chosen to be the elements in $\mathbb{Z} \cap (-q/2, q/2]$. Let $\mathbb{Z}_q[X] \coloneqq (\mathbb{Z}/q\mathbb{Z})[X]$ be the ring of polynomials over the ring $\mathbb{Z}_q$. Elements of $\mathbb{Z}_q[X]$ are polynomials of the form $\sum_{j=0}^{D} a_j X^j$ for some $D \in \mathbb{N}$ where $a_j \in \mathbb{Z}_q$ for $j = 0, 1, \ldots, D$. Let $\mathbb{Z}[X]$ and $\mathbb{R}[X]$ be the ring of polynomials over the ring $\mathbb{Z}$ and $\mathbb{R}$, respectively. Elements of $\mathbb{Z}[X]$ and $\mathbb{R}[X]$ are polynomials of the form $\sum_{j=0}^{D} a_j X^j$ for some $D \in \mathbb{N}$ where $a_j \in \mathbb{Z}$ and $a_j \in \mathbb{R}$, respectively, for $j = 0, 1, \ldots, D$.

For a power-of-2 $N \in \mathbb{N}$, let $\Phi_{2N}(X)$ be the $(2N)$-th cyclotomic polynomial, of degree N, which takes the form $\Phi_{2N}(X) = X^N + 1$. Let $\mathcal{R} \coloneqq \mathbb{Z}[X]/(\Phi_{2N}(X))$ and $\mathcal{R}_q \coloneqq \mathbb{Z}_q[X]/(\Phi_{2N}(X))$ be the ring of polynomials $\mathbb{Z}[X]$ and $\mathbb{Z}_q[X]$, respectively, modulo polynomial $\Phi_{2N}(X)$. Polynomial modulo is understood to be the remainder polynomial term after polynomial long division. Elements of $\mathcal{R}$ and $\mathcal{R}_q$ are polynomials of the form $\sum_{j=0}^{N-1} a_j X^j$ where $a_j \in \mathbb{Z}$ and $a_j \in \mathbb{Z}_q$, respectively, for $j = 0, 1, \ldots, N-1$. Let $\mathcal{S} \coloneqq \mathbb{R}[X]/(\Phi_{2N}(X))$ be the ring of polynomials $\mathbb{R}[X]$ modulo polynomial $\Phi_{2N}(X)$. Elements of $\mathcal{S}$ are polynomials of the form $\sum_{j=0}^{N-1} a_j X^j$ where $a_j \in \mathbb{R}$ for $j = 0, 1, \ldots, N-1$.

Example 1 (Rings)**.** *$\mathbb{Z}_q$. Let $\overline{a}, \overline{b} \in \mathbb{Z}_q$. For $q \coloneqq 6$, $\mathbb{Z}_6 = \{\overline{-2}_6, \overline{-1}_6, \overline{0}_6, \overline{1}_6, \overline{2}_6, \overline{3}_6\}$. The ring operations are*

- *Addition: $\overline{a}+\overline{b} \coloneqq (a+b) \mod q$, e.g. $\overline{-2}_6+\overline{-1}_6 = (-3) \mod 6 = \overline{3}_6$. The additive identity is $\overline{0}_q$, e.g. $\overline{0}_6$. The additive inverse is $-\overline{a} \coloneqq (-a) \mod q$, e.g. $-\overline{-2}_6 = 2 \mod 6 = \overline{2}_6$.*
- *Multiplication: $\overline{a} \cdot \overline{b} \coloneqq (a \cdot b) \mod q$, e.g. $\overline{-2}_6 \cdot \overline{-1}_6 = 2 \mod 6 = \overline{2}_6$. The multiplicative identity is $\overline{1}_q$, e.g. $\overline{1}_6$.*

$\mathbb{Z}_q[X]$. For $a \coloneqq \sum_{j=0}^{D_a} a_j X^j \in \mathbb{Z}_q[X]$ and $b \coloneqq \sum_{j=0}^{D_b} b_j X^j \in \mathbb{Z}_q[X]$, the ring operations are

- *Addition: $a+b \coloneqq \sum_{j=0}^{\max\{D_a, D_b\}} (a_j + b_j) X^j$, where $a_j + b_j$ is addition as defined for $\mathbb{Z}_q$, and $a_j = b_i \coloneqq \overline{0}_q$ for $j > D_a$, $i > D_b$. E.g. $a \coloneqq \overline{-2}_6 + \overline{3}_6 X$, $b \coloneqq \overline{-1}_6 \implies a + b = \overline{3}_6 + \overline{3}_6 X$. The additive identity is $\overline{0}_q$, e.g. $\overline{0}_6$. The additive inverse is $-a \coloneqq \sum_{j=0}^{D_a} (-a_j) X^j$, where $-a_j$ is as defined for $\mathbb{Z}_q$. E.g. $a \coloneqq \overline{-2}_6 + \overline{3}_6 X \implies -a = \overline{2}_6 + \overline{3}_6 X$.*
- *Multiplication: $a \cdot b \coloneqq \sum_{j=0}^{D_a+D_b} c_j X^j$ for $c_j \coloneqq \sum_{i=0}^{j} a_i \cdot b_{j-i}$, where $a_i \cdot b_{j-i}$ is as multiplication defined for $\mathbb{Z}_q$. E.g. $a \coloneqq \overline{-2}_6 + \overline{3}_6 X$, $b \coloneqq \overline{-2}_6 \implies a \cdot b = \overline{2}_6 + \overline{0}_6 X = \overline{2}_6$.*

$\mathcal{R}_q$. For $N \coloneqq 2$, $\Phi_{2N} = X^2 + 1$. For $a \coloneqq \sum_{j=0}^{D_a} a_j X^j \in \mathcal{R}_q$ and $b \coloneqq \sum_{j=0}^{D_b} b_j X^j \in \mathcal{R}_q$, with $D_a, D_b \leq N-1$, the ring operations are

- *Addition:* $a+b \coloneqq \left(\sum_{j=0}^{\max\{D_a,D_b\}}(a_j+b_j)X^j\right) \mod (X^N+1)$, *where* a_j+b_j *is addition as defined for* $\mathbb{Z}_q$, *and* $a_j = b_i \coloneqq \overline{0}_q$ *for* $j > D_a,\ i > D_b$. *E.g.* $a \coloneqq \overline{-2}_6+\overline{3}_6X,\ b \coloneqq \overline{-1}_6 \implies a+b = (\overline{3}_6+\overline{3}_6X) \mod (X^2+1) = \overline{3}_6+\overline{3}_6X$. *The additive identity is* $\overline{0}_q$, *e.g.* $\overline{0}_6$. *The additive inverse is* $-a \coloneqq \left(\sum_{j=0}^{D_a}(-a_j)X^j\right) \mod (X^N+1)$, *where* $-a_j$ *is as defined for* $\mathbb{Z}_q$. *E.g.* $a \coloneqq \overline{-2}_6+\overline{3}_6X \implies -a = (\overline{2}_6+\overline{3}_6X) \mod (X^2+1)$. *Note that addition cannot lead to a larger degree, hence the (.* $\mod (X^N+1)$*)-operation may be dropped.*
- *Multiplication:* $a \cdot b \coloneqq \left(\sum_{j=0}^{D_a+D_b} c_jX^j\right) \mod (X^N+1)$, *for* $c_j \coloneqq \sum_{i=0}^{j} a_i \cdot b_{j-i}$, *where* $a_i \cdot b_{j-i}$ *is multiplication as defined for* $\mathbb{Z}_q$. *E.g.* $a \coloneqq \overline{-2}_6+\overline{-1}_6X,\ b \coloneqq \overline{2}_6X \implies a \cdot b = (\overline{2}_6X+\overline{-2}_6X^2) \mod (X^2+1) = [\overline{-2}_6(X^2+1)+(-\overline{-2}_6)+\overline{2}_6X] \mod (X^2+1) = \overline{2}_6+\overline{2}_6X$.

$\mathbb{Z}[X]$, $\mathcal{R}$, $\mathbb{R}[X]$, *and* $\mathcal{S}$ *behaves as* $\mathbb{Z}_q[X]$ *and* $\mathcal{R}_q$, *respectively, except that the polynomial coefficients are integers and real numbers so addition, and multiplication acts accordingly.*

Encoding/Decoding Fix a power-of-2 $N \in \mathbb{N}$. Let $\mathbb{Z}_{2N}^* \coloneqq \{x \in \mathbb{Z}_{2N} : \gcd(x, 2N) = 1\}$ be the set of elements in $\mathbb{Z}_{2N}$, the ring of integers modulo $2N$, that are coprimes with $2N$. Let T be a subgroup of $\mathbb{Z}_{2N}^*$ such that $\mathbb{Z}_{2N}^*/T = \{\pm 1\}$. It follows that $|T| = N/2$. T serves as the index set of the complex vector to be encoded. Let $\mathbb{H} \coloneqq \{(z_j)_{j\in\mathbb{Z}_{2N}^*} : z_j \in \mathbb{C}, z_{-j} = \bar{z}_j\ \forall j \in \mathbb{Z}_{2N}^*\} \subset \mathbb{C}^N$ be the set of N-large complex vectors whose element indexed by $(-j)$ is the complex conjugate of its element indexed by j. Let $\zeta_{2N} \coloneqq \exp(-\pi i/N) \in \mathbb{C}$ for the imaginary unit i.

Decoding maps a polynomial $m(X) \in \mathcal{R}$ to a complex vector $\boldsymbol{z} \coloneqq (z_j)_{j\in T} \in \mathbb{C}^{N/2}$ by $\boldsymbol{z} = (\pi \circ \sigma)(m)$. $\sigma : \mathcal{R} \to \mathbb{C}^N$ is defined as $\sigma(m) \coloneqq (m(\zeta_{2N}^j))_{j\in\mathbb{Z}_{2N}^*}$, the evaluation values of the polynomial $m(X)$ at j-th powers of ζ_{2N}. $\pi : \mathbb{H} \to \mathbb{C}^{N/2}$ sends the evaluation values $(m(\zeta_{2N}^j))_{j\in\mathbb{Z}_{2N}^*}$ to the vector $\boldsymbol{z} = (z_j)_{j\in T}$.

Encoding maps a complex vector $\boldsymbol{z} = (z_j)_{j\in T} \in \mathbb{C}^{N/2}$ to a polynomial in $m(X) \in \mathcal{R}$ by $\sigma^{-1}\left(\lfloor \pi^{-1}(\boldsymbol{z}) \rceil_{\sigma(\mathcal{R})}\right)$, where $\lfloor . \rceil_{\sigma(\mathcal{R})}$ means rounding to an element in $\sigma(\mathcal{R})$. Where $\sigma^{-1} : \sigma(\mathcal{R}) \to \mathcal{R}$ is the inverse of σ, and $\pi^{-1} : \mathbb{C}^{N/2} \to \mathbb{H}$, inverse of π, is given by $(\pi^{-1}(\boldsymbol{z}))_j = z_j$ if $j \in T$, $\bar{z}_{-j}$ otherwise, for $j \in \mathbb{Z}_{2N}^*$. Encoding is almost the inverse of decoding except the rounding step. Because of the precision loss in rounding, $\boldsymbol{z}$ is usually scaled by a power-of-2 $\Delta > 2^0$. The encoding/decoding algorithms are isometric homomorphisms between metric spaces $(\mathcal{S}, ||.||_\infty^{\mathsf{can}})$ and $(\mathbb{C}^{N/2}, ||.||_\infty)$ for the norms $||(z_j)_{j\in T}||_\infty \coloneqq \max_{j\in T} |z_j|$ and $||a||_\infty^{\mathsf{can}} \coloneqq ||\sigma(a)||_\infty$ for $a \in \mathcal{S}$.

Example 2 (Encoding, Cheon et al. (2017) p. 418)**.** *Let* $N \coloneqq 4$, *so* $\mathbb{Z}_{2N} = \{\overline{-3}_8, \overline{-2}_8, \ldots, \overline{4}_8\}$ *and* $\mathbb{Z}_{2N}^* = \{-3, -1, 1, 3\}$ *(for simplicity, I drop the* $\bar{a}_8$ *notation). Set* $T \coloneqq \{1, 3\}$, *so we have the vector* $\boldsymbol{z} = (z_1, z_3)$. *Let* $\boldsymbol{z} = (z_1, z_3) \coloneqq (3+4i, 2-i) \in \mathbb{C}^2$, *and the scale factor* $\Delta \coloneqq 64$. *Then* $\pi^{-1}(\boldsymbol{z}) = (2+i, 3-4i, 3+4i, 2-i) \in \mathbb{C}^4$. *Using* σ^{-1}, *we fit a polynomial* $\hat{m}(X)$ *such that* $(\hat{m}(\zeta_{2N}^j))_{j\in\mathbb{Z}_{2N}^*} = \Delta \cdot \boldsymbol{z}$. *In practice, the polynomial-fitting can be done with the Lagrange-polynomial or saturated ordinary least squares. The latter scales better with* N *exploiting orthogonality in the design matrix. The fitted polynomial, suppressing some decimals, is* $160 - 45.2548X - 159.9999X^2 - 90.5097X^3$. *Performing coefficient-wise rounding, the output*

of the encoding algorithm is $m(X) = 160 - 45X - 160X^2 - 91X^3$*. At* $\zeta_{2N}^1, \zeta_{2N}^3$*, the polynomial* $\Delta^{-1}m(X)$ *has evaluation values* $(3.0082 + 4.0026i, 1.9918 - 0.9974i) \approx \boldsymbol{z}$*, which is the output of the decoding algorithm.*

Probability Distributions For $\sigma \in \mathbb{R}_{>0}$, let $\mathcal{DG}(\sigma^2)$ be the univariate discrete Gaussian distribution of mean 0 and variance σ^2 (see Karney (2016) for description and sampling). For positive integer $H \leq N$, let $\mathcal{HWT}(H) \coloneqq \{\boldsymbol{x} \in \{-1, 0, 1\}^N : \sum_{i=1}^N \mathbb{1}_{x_i \neq 0} = H\}$ be the set of N-large vectors with entries in $\{-1, 0, 1\}$ with exactly H non-zero entries. For $\rho \in [0, 1]$, let $\mathcal{ZO}(\rho)$ be a distribution on vectors $\boldsymbol{x} \in \{-1, 0, 1\}^N$ such that entries x_i of $\boldsymbol{x}$ are independently and identically distributed with $\mathbb{P}(x_i = -1) = \mathbb{P}(x_i = 1) = \rho/2$ and $\mathbb{P}(x_i = 0) = 1 - \rho$ for $i = 1, 2, \ldots, N$. $y \leftarrow D$ denotes sampling a polynomial y (by sampling its coefficient vector) from D where D can be either (i) a distribution like $\mathcal{DG}(\sigma^2)$ or $\mathcal{ZO}(\rho)$; or (ii) a finite set like $\mathcal{HWT}(H)$ or $\mathcal{R}_q$. When D is a set, $\leftarrow$ denotes sampling uniformly from D. When D is $\mathcal{DG}(\sigma^2)$, each entry of the coefficient vector of the polynomial is sampled independently from $\mathcal{DG}(\sigma^2)$.

2.1.3 Complete Scheme

The CKKS scheme (Cheon et al. (2017), Cheon et al. (2018)) is uniquely characterised by six parameters: $N, \lambda, L, P, H, \sigma$. The power-of-2 N governs the degree of the polynomials as above. $\lambda \in \mathbb{R}$ is the security level, meaning that every known attack against the encryption scheme takes $\Omega(2^\lambda)$ bit operations (Cheon et al., 2017). $L \in \mathbb{N}$, the initial level of ciphertexts, is governed by precision (Δ) and the multiplicative depth of the algorithm ($D_{\mathsf{(C)Mult}}$, see below). Specifically, $L \geq (D_{\mathsf{(C)Mult}} + 1)\log_2 \Delta$ must be to obtain $\log_2 \Delta$ bits of precision. Define the sequence $q_\ell \coloneqq p^\ell q_0$ for $p, q_0 \in \mathbb{N}$ and the level $\ell \in \mathbb{N} \cap (0, L]$ of ciphertexts. $P \in \mathbb{N}$ is an auxiliary parameter.

The parameters are related by the RLWE security estimate $N \geq \frac{\lambda+110}{7.2}\log_2(Pq_L)$, see Cheon et al. (2017). Following Kim et al. (2018a), I set $q_0 \coloneqq 1, p \coloneqq 2, P \coloneqq q_L$, therefore

$$N \geq \frac{\lambda + 110}{7.2} \cdot 2L \iff \frac{7.2N}{2L} - 110 \geq \lambda. \tag{1}$$

This has severe implications. When we increase the complexity of the homomorphic algorithm (larger $D_{\mathsf{(C)Mult}}$) or need more precision (larger Δ), L increases. If the increase in L is not offset by an increase in N, the encryption security λ decreases. That is, more complex or precise algorithms require larger N and thus longer computation time to guarantee the same level of security. As in Cheon et al. (2017), I set $H \coloneqq 64$, and, following common practice (Albrecht et al., 2018), $\sigma \coloneqq 8/\sqrt{2\pi}$.

The complete scheme consists of the following algorithms.

- $\mathsf{KeyGen}()$ (key generation). For parameters $N, \lambda, L, P, H, \sigma$ as discussed above:
  - Sample polynomials $s \leftarrow \mathcal{HWT}(H)$, $a \leftarrow \mathcal{R}_{q_L}$ and $e \leftarrow \mathcal{DG}(\sigma^2)$. Set the secret key $\mathsf{sk} \coloneqq (1, s) \in \mathcal{R}_{q_L}^2$, the public key $\mathsf{pk} \coloneqq (b, a) \in \mathcal{R}_{q_L}^2$ with $b \coloneqq -as + e \pmod{q_L}$.

– Sample polynomials $a' \leftarrow \mathcal{R}_{Pq_L}$, $e' \leftarrow \mathcal{DG}(\sigma^2)$. Set the evaluation key $\mathsf{evk} \coloneqq (b', a') \in \mathcal{R}^2_{Pq_L}$ with $b' \coloneqq -a's + e' + Ps^2 \pmod{Pq_L}$.

Output the keys $(\mathsf{sk}, \mathsf{pk}, \mathsf{evk})$.

- $\mathsf{KSGen}_{\mathsf{sk}}(s')$ (switching key generation). For secret key $\mathsf{sk} = (1, s) \in \mathcal{R}^2_{q_L}$ and $s' \in \mathcal{R}$, sample polynomials $a' \leftarrow \mathcal{R}_{Pq_L}$, $e' \leftarrow \mathcal{DG}(\sigma^2)$. Output the switching key $\mathsf{swk} \coloneqq (b', a') \in \mathcal{R}^2_{Pq_L}$ with $b' \coloneqq -a's + e' + Ps' \pmod{Pq_L}$.
- $\mathsf{Ecd}_\Delta(\boldsymbol{z})$ (encoding). For $\boldsymbol{z} = (z_j)_{j\in T} \in \mathbb{C}^{N/2}$ and scale Δ, output $\sigma^{-1}\left(\lfloor \Delta \pi^{-1}(\boldsymbol{z}) \rceil_{\sigma(\mathcal{R})}\right)$, the encoding of $\boldsymbol{z}$.
- $\mathsf{Dcd}_\Delta(m)$ (decoding). For polynomial $m(X) \in \mathcal{R}$, output $\Delta^{-1}(\pi \circ \sigma)(m) \in \mathbb{C}^{N/2}$, the decoding of $m(X)$.
- $\mathsf{Enc}_{\mathsf{pk}}(m)$ (encryption). For polynomial $m(X) \in \mathcal{R}$ and public key $\mathsf{pk} = (b, a) \in \mathcal{R}^2_{q_L}$, sample polynomials $v \leftarrow \mathcal{ZO}(0.5)$ and $e_0, e_1 \leftarrow \mathcal{DG}(\sigma^2)$. Output the ciphertext $\boldsymbol{c} \coloneqq (c_0, c_1) \in \mathcal{R}^2_{q_L}$, the encryption of m, with $c_0 \coloneqq vb + m + e_0 \pmod{q_L}$ and $c_1 \coloneqq va + e_1 \pmod{q_L}$.
- $\mathsf{Dec}_{\mathsf{sk}}(\boldsymbol{c})$ (decryption). For ciphertext $\boldsymbol{c} = (c_0, c_1) \in \mathcal{R}^2_{q_\ell}$ at level ℓ and secret key $\mathsf{sk} = (1, s) \in \mathcal{R}^2_{q_L}$, output $m \coloneqq c_0 + c_1 s \pmod{q_\ell}$, the decryption of $\boldsymbol{c}$.
- $\mathsf{Add}(\boldsymbol{c}_1, \boldsymbol{c}_2)$ (addition). For two ciphertexts $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{R}^2_{q_\ell}$, output $\boldsymbol{c} \coloneqq \boldsymbol{c}_1 + \boldsymbol{c}_2 \pmod{q_\ell} \in \mathcal{R}^2_{q_\ell}$ the homomorphic sum of $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$.
- $\mathsf{CAdd}(\boldsymbol{c}, a)$ (addition with constant). For ciphertext $\boldsymbol{c} \in \mathcal{R}^2_{q_\ell}$, and a constant $a \in \mathcal{R}$, output $\boldsymbol{c} \coloneqq \boldsymbol{c} + (a, 0) \pmod{q_\ell} \in \mathcal{R}^2_{q_\ell}$ the homomorphic sum of $\boldsymbol{c}$ and a.
- $\mathsf{Mult}_{\mathsf{evk}}(\boldsymbol{c}_1, \boldsymbol{c}_2)$ (multiplication). For two ciphertexts $\boldsymbol{c}_1 = (b_1, a_1) \in \mathcal{R}^2_{q_\ell}, \boldsymbol{c}_2 = (b_2, a_2) \in \mathcal{R}^2_{q_\ell}$ and the evaluation key evk, set $(d_0, d_1, d_2) \coloneqq (b_1 b_2, a_1 b_2 + a_2 b_1, a_1 a_2) \pmod{q_\ell} \in \mathcal{R}^3_{q_\ell}$. Output $\boldsymbol{c} \coloneqq (d_0, d_1) + \lfloor P^{-1} d_2 \cdot \mathsf{evk} \rceil \pmod{q_\ell} \in \mathcal{R}^2_{q_\ell}$, the homomorphic product of $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$.
- $\mathsf{CMult}(\boldsymbol{c}, a)$ (multiplication with constant). For ciphertext $\boldsymbol{c} \in \mathcal{R}^2_{q_\ell}$, and a constant $a \in \mathcal{R}$, output $\boldsymbol{c} \coloneqq a \cdot \boldsymbol{c} \pmod{q_\ell} \in \mathcal{R}^2_{q_\ell}$, the homomorphic product of $\boldsymbol{c}$ and a.
- $\mathsf{RS}_{\ell \to \ell'}(\boldsymbol{c})$ (rescaling). For ciphertext $\boldsymbol{c} \in \mathcal{R}^2_{q_\ell}$ at level ℓ and the new level $\ell' \in \mathbb{N} \cap (0, \ell]$, output $\boldsymbol{c}' \coloneqq \left\lfloor \frac{q_{\ell'}}{q_\ell} \boldsymbol{c} \right\rceil \pmod{q_{\ell'}} \in \mathcal{R}^2_{q_{\ell'}}$, the rescaled $\boldsymbol{c}$. Subscript $\ell \to \ell'$ is omitted for $\ell' = \ell - \log_2 \Delta$.
- $\mathsf{ModDown}(\boldsymbol{c}, q_{\ell'})$ (modulus reduction). For ciphertext $\boldsymbol{c} = (c_0, c_1) \in \mathcal{R}^2_{q_\ell}$ at level ℓ and new modulus $q_{\ell'}$ with $\ell' \le \ell$, output $\boldsymbol{c'} \coloneqq (c_0 \mod q_{\ell'}, c_1 \mod q_{\ell'}) \in \mathcal{R}^2_{q_{\ell'}}$.
- $\mathsf{KS}_{\mathsf{swk}}(\boldsymbol{c'})$ (key switching). For ciphertext $\boldsymbol{c'} = (c'_0, c'_1) \in \mathcal{R}^2_{q_\ell}$ encrypting $m \in \mathcal{R}$ under secret key $\mathsf{sk}' = (1, s') \in \mathcal{R}^2_{q_L}$ and $\mathsf{swk} = \mathsf{KSGen}_{\mathsf{sk}}(s')$ output $\boldsymbol{c} \coloneqq (c'_0, 0) + \lfloor P^{-1} c'_1 \cdot \mathsf{swk} \rceil \pmod{q_\ell} \in \mathcal{R}^2_{q_\ell}$ encrypting m under secret key $\mathsf{sk} = (1, s) \in \mathcal{R}^2_{q_L}$.
- $\mathsf{Perm}^{i \to j}_{\mathsf{swk}_{i,j}}(\boldsymbol{c})$ (permutation). For ciphertext $\boldsymbol{c} = (c_0, c_1) \in \mathcal{R}^2_{q_\ell}$, encrypting (the encoding of) $(z_j)_{j\in T}$ under secret key $\mathsf{sk}(1, s) \in \mathcal{R}^2_{q_L}$, indices $i, j \in T$, switching key $\mathsf{swk}_{i,j} =$

$\mathsf{KSGen}(\kappa_k(\mathsf{sk}))$ where $\kappa_k : m(X) \mapsto m(X^k) \pmod{(X^N+1)}$ with $k \coloneqq j^{-1}i \pmod{2N}$, [2] $\kappa_k(\mathsf{sk}) \coloneqq \kappa_k(s)$, let $\boldsymbol{c'} \coloneqq (\kappa_k(c_0), \kappa_k(c_1)) \in \mathcal{R}^2_{q_\ell}$. Output $\mathsf{KS}_{\mathsf{swk}_{i,j}}(\boldsymbol{c'}) \in \mathcal{R}^2_{q_\ell}$, encrypting (the encoding of) $(\hat{z}_j)_{j\in T}$ under secret key $\mathsf{sk} = (1, s) \in \mathcal{R}^2_{q_L}$ such that $\hat{z}_j = z_i$ and $\hat{z}_{j'} = z_{j'},\ \forall j' \in T : j' \neq j$.

- $\mathsf{Sum}_{(\mathsf{swk}_{i,j})_{i,j\in T}}(\boldsymbol{c})$ (vector-summation) For ciphertext $\boldsymbol{c} \in \mathcal{R}^2_{q_\ell}$, encrypting (the encoding of) $(z_j)_{j\in T}$, and switching keys $(\mathsf{swk}_{i,j})_{i,j\in T}$ where $\mathsf{swk}_{i,j} \coloneqq \mathsf{KSGen}(\kappa_k(\mathsf{sk}))$ with $k \coloneqq j^{-1}i \pmod{2N}$, output $\boldsymbol{c}_s \in \mathcal{R}^2_{q_\ell}$, encrypting (the encoding of) $(\hat{z}_j)_{j\in T}$, the first element of which is $\sum_{j\in T} z_j$. $\boldsymbol{c}_s$ is computed by Algorithm 4.

Example 3 illustrates the algorithms of the scheme. Some remarks are due: (i) polynomials referred to as *constants* because they are not encrypted (so anyone can access the associated data via Dcd), not because they are of the form $a(X) = a_0$ for some scalar a_0; and (ii) $D_{\mathsf{(C)Mult}}$, the multiplicative depth of some algorithm (any composition of the above algorithms), is the largest number of multiplications (Mult or CMult) performed on the ciphertext involved in the most multiplications. E.g. if there are three ciphertexts $\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3$ in an algorithm – each of them is at level L before any computation is performed on them – and they are involved in 23, 3, 26 multiplications (Mult or CMult) followed by rescaling (RS), respectively, then $D_{\mathsf{(C)Mult}} = 26$.

Example 3 (CKKS)**.** *Let $N \coloneqq 4$ so that $\mathbb{Z}_{2N} = \{-3,-2,-1,0,1,2,3,4\}$, $\mathbb{Z}^*_{2N} = \{-3,-1,1,3\}$, and set $T \coloneqq \{1,3\}$. Let $\boldsymbol{z}_A = (z_{A,1}, z_{A,3}) \coloneqq (3+4i, 2-i) \in \mathbb{C}^2$, $\boldsymbol{z}_B = (z_{B,1}, z_{B,3}) \coloneqq (2, 10) \in \mathbb{C}^2$, and $(\mathsf{sk}, \mathsf{pk}, \mathsf{evk}) \coloneqq \mathsf{KeyGen}()$. Then CKKS exhibits the following behavior.*

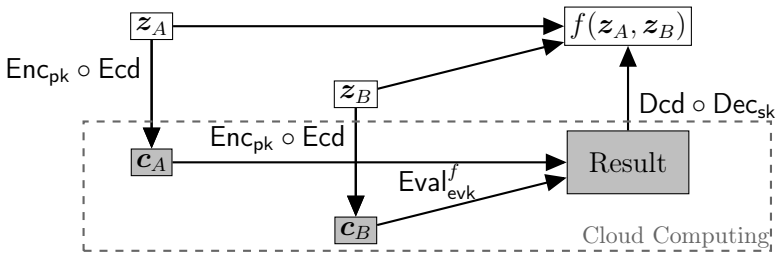
- *Encoding/Decoding. $\mathsf{Dcd}_\Delta(\mathsf{Ecd}_\Delta(\boldsymbol{z}_A)) \approx \boldsymbol{z}_A$.*
- *Encryption/Decryption. $\mathsf{Dcd}_\Delta(\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(\boldsymbol{z}_A)))) \approx \boldsymbol{z}_A$.*
- *Addition. Let $\boldsymbol{c}_A \coloneqq \mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(\boldsymbol{z}_A))$, $\boldsymbol{c}_B \coloneqq \mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(\boldsymbol{z}_B))$, and $\boldsymbol{c}_{add} \coloneqq \mathsf{Add}(\boldsymbol{c}_A, \boldsymbol{c}_B)$. Then $\mathsf{Dcd}_\Delta(\mathsf{Dec}_{\mathsf{sk}}(\boldsymbol{c}_{add})) \approx \boldsymbol{z}_A + \boldsymbol{z}_B = (5+4i, 12-i)$.*
- *Addition with constant. Let $\boldsymbol{c}_A \coloneqq \mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(\boldsymbol{z}_A))$, and $a \coloneqq \mathsf{Ecd}_\Delta((100, 10))$. Then $\mathsf{Dcd}_\Delta(\mathsf{Dec}_{\mathsf{sk}}(\mathsf{CAdd}(\boldsymbol{c_A}, a))) \approx (103+4i, 12-i)$.*
- *Multiplication. Let $\boldsymbol{c}_A \coloneqq \mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(\boldsymbol{z}_A))$, $\boldsymbol{c}_B \coloneqq \mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(\boldsymbol{z}_B))$, and let $\boldsymbol{c}_{mult} \coloneqq \mathsf{Mult}_{\mathsf{evk}}(\boldsymbol{c}_A, \boldsymbol{c}_B)$. Then $\mathsf{Dcd}_{\Delta^2}(\mathsf{Dec}_{\mathsf{sk}}(\boldsymbol{c}_{mult})) \approx \boldsymbol{z}_A \cdot \boldsymbol{z}_B = (6+8i, 20-10i)$.*
- *Multiplication with constant. Let $\boldsymbol{c}_A \coloneqq \mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(\boldsymbol{z}_A))$, and $a \coloneqq \mathsf{Ecd}_\Delta((100, 10))$. Then $\mathsf{Dcd}_{\Delta^2}(\mathsf{Dec}_{\mathsf{sk}}(\mathsf{CMult}(\boldsymbol{c_A}, a)) \approx (300+400i, 20-10i)$.*
- *Rescaling. RS, used after multiplications, plays an important role in dynamically managing the size of ciphertexts. For more details see Cheon et al. (2017).*
- *Modulus reduction. $\mathsf{ModDown}$ makes it possible to add ciphertexts with different modulus by reducing the modulus of the ciphertext with higher modulus. For more details see Cheon et al. (2017).*

[2] $j^{-1} \in \mathbb{Z}^*_{2N}$ is well-defined because $\mathbb{Z}^*_{2N}$ is a multiplicative group and T is its subgroup.

- *Permutation. Let* $\boldsymbol{c}_A \coloneqq \mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(\boldsymbol{z}_A))$ *be the encryption of* $\boldsymbol{z}_A = (z_{A,1}, z_{A,3})$ *under secret key* sk*. By applying the permutation algorithm to* $\boldsymbol{c}_A$ *we can get a ciphertext* $\mathsf{Perm}^{i\to j}_{\mathsf{swk}_{i,j}}(\boldsymbol{c}_A)$ *which is the encryption of* $\boldsymbol{z}'_A = (z'_{A,1}, z'_{A,3})$*, where* $z'_{A,j} = z_{A,i}$*, under secret key* sk*. That is, we send element* i *to element* j*. For instance, let* $i \coloneqq 1$ *and* $j \coloneqq 3$*. Then* $\mathsf{Dcd}_\Delta(\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Perm}^{1\to 3}_{\mathsf{swk}_{1,3}}(\boldsymbol{c}_A))) \approx (3 + 4i, 3 + 4i)$*. Another example is* $\mathsf{Dcd}_\Delta(\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Perm}^{3\to 1}_{\mathsf{swk}_{3,1}}(\boldsymbol{c}_A))) \approx (2 - i, 2 - i)$*.*
- *Vector-summation. Let* $\boldsymbol{c} \coloneqq \mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(\boldsymbol{z}))$ *be the encryption of* $\boldsymbol{z} = (z_1, z_3)$ *under secret key* sk*. Then for* $\boldsymbol{c}_s \coloneqq \mathsf{Sum}_{(\mathsf{swk}_{i,j})_{i,j\in T}}(\boldsymbol{c})$*, the first element of* $\mathsf{Dcd}_\Delta(\mathsf{Dec}_{\mathsf{sk}}(\boldsymbol{c}_s))$ *is approximately* $z_1 + z_3$*.*

Visual illustration. Let f *be some composition of addition and multiplication, and* $\mathsf{Eval}^f_{\mathsf{evk}}$ *be the same composition of* $\mathsf{Add}, \mathsf{CAdd}$ *and* $\mathsf{Mult}_{\mathsf{evk}}, \mathsf{CMult}$*. Then Figure 2 depicts that we can obtain* $f(\boldsymbol{z}_A, \boldsymbol{z}_B)$ *with homomorphic encryption. That is, Figure 2 is a commutative diagram.*

Figure 2: Operations with Homomorphic Encryption



Notes: in shaded nodes data are encrypted.

2.2. Applications in Statistics

HE enables statistical queries without sacrificing privacy. However, it suffers from a major drawback beside large computational time. Namely, homomorphic encryption schemes only support addition ($\mathsf{Add}, \mathsf{CAdd}$) and multiplication ($\mathsf{Mult}, \mathsf{CMult}$) natively. This can be resolved by bit-wise encryption, but such procedures are much slower (Song et al., 2019).[3] Hence special algorithms need to be designed, which led to active research since the breakthrough of Gentry (2009).

Yang et al. (2019) provide a comprehensive survey of HE applications, ranging from elementary operations on sets and matrices to more complex algorithms in statistics, machine learning and computer science. Below I list several examples from the survey (and additional works) on statistics and machine learning.

[3]Simply put, the intuition is that today's (non-quantum) computers represent a number with bits, and each operation can be described by logical gates which boil down to addition (OR) or multiplication (AND).

HE has been used for hypothesis testing. Lu et al. (2015) propose a way for computing the χ^2 statistics for testing independence between nominal variables, safely aggregating data from multiple clients. Poon et al. (2018) give an algorithm for Fisher's exact test. However, for both tests, the last step, division, is performed by the client. On the contrary, Zhang et al. (2015) use an encrypted look-up table procedure to "perform division" on encrypted *integers* to compute the χ^2 statistics.

Linear and logistic regressions are more complex HE applications because they involve matrix inversion or optimisation. Optimisation is typically managed by iterative gradient methods, mostly with low number of iterations. For example, Esperança et al. (2017) estimate a linear (ridge) regression with an accelerated gradient method using the BFV scheme (Fan and Vercauteren, 2012). Logistic regression received significant attention, partly thanks to the iDash 2017 competition.[4] Kim et al. (2018a), in an improved version of Kim et al. (2018b), use the CKKS scheme for logistic regression. In both works, the exponential function of the logit function is approximated with least squares by a polynomial. On the other hand, Yoo et al. (2019) use bit-wise encryption to compute (more) exact exponential function.

Regarding non-parametric methods, HE appears to have been used mostly for classification. Graepel et al. (2012) propose methods for encrypted classification, but the encoding routine is not as efficient as that of CKKS, and it is the client's task to assign the classification label from some score in the end. Bost et al. (2015) provide a framework for classification including Support Vector Machines (SVM), Naïve Bayes (NB) and Decision Tree (DT). They design algorithms to compare $(\leq, \geq)$ encrypted data and support operations like $\arg\max$ to homomorphically assign the classification label. However, comparisons require communication between the client and the server, and their encoding/encryption method are not as efficient as that of CKKS.

Khedr et al. (2015) implement the NB classifier with the GSW Gentry et al. (2013) scheme. Bian et al. (2019), using the Paillier scheme (Paillier, 1999) with bit-wise encryption, apply the NB classifier to e-mail spam filtering.

Alabdulkarim et al. (2019) fit a DT where data are hold by multiple parties and encrypted summary statistics are shared between them. The fitting happens simultaneously and iteratively wherein each round parties share summary statistics. Nonetheless, the fitting itself takes place in the clean (i.e. on non-encrypted data).

González-Serrano et al. (2018) fit SVM but operations not natively supported by HE are carried out by a trusted third party. Rahulamathavn et al. (2014) train SVM using only polynomial kernels to avoid exponential functions. Park et al. (2020) use CKKS to fit SVM with polynomial and radial basis function kernel, which involves the exponential function.

[4] http://www.humangenomeprivacy.org/2017/

3. Kernel Density Estimation

In this section, Kernel Density Estimation (KDE) and its difficulties when applied with HE are discussed. Next, I devise a KDE method designed for HE, called HE-KDE. It consists of two parts: the estimator on non-encrypted data, and algorithms enabling the use of the estimator on encrypted data. Last, the properties of HE-KDE on non-encrypted data are derived and optimal parameter choices are discussed.

3.1. Description

We observe an n-large sample $\boldsymbol{x} \coloneqq (x_i)_{i\in[n]} \in \mathbb{R}^n$, $[n] \coloneqq 1, 2, \ldots, n$, with elements x_i independently and identically distributed (*i.i.d.*) deviates of the real-valued random variable X, which admits a density $f : \mathbb{R} \to \mathbb{R}_+$. Throughout, I assume that f has bounded support $S \coloneqq [a, b] \subset \mathbb{R}$, $a < b$. Equivalently, $f(x) = 0$ if $x \notin S$. The objective of nonparametric density estimation is to estimate f from the sample $\boldsymbol{x}$ without imposing a functional form on f. KDE is a nonparametric density estimator that uses kernel functions. It takes the form $\hat{f}(x) \coloneqq \frac{1}{nh}\sum_{i\in[n]} K\left(\frac{x-x_i}{h}\right)$ for bandwidth $h \in \mathbb{R}_{++}$ and kernel function $K : \mathbb{R} \to \mathbb{R}_+$. K is a weighting function which puts smaller weights on distant observations (as $|x - x_i| \to \infty$, the weight $K((x - x_i)/h) \to 0$). The bandwidth h governs how sensitive the down-weighting is to the difference $x - x_i$ (as $h \to 0$, even the distance between closest observations are enlarged, causing $K((x - x_i)/h) \to 0$). Large sensitivity (small h) implies rough $\hat{f}$ (large variance, small bias), while small sensitivity (large h) implies smooth $\hat{f}$ (small variance, large bias). One chooses the bandwidth to balance variance and bias.

3.2. HE Difficulties

For non-encrypted KDE, the choice of kernel is generally not very important (Pagan and Ullah, 1999). However, the kernel choice becomes intricate with HE. To see why, note that typically used kernels either have (i) bounded support (e.g. $K(u) = 1/2$ if $u \in [\pm 1]$, 0 otherwise); or (ii) unbounded support and a nonlinear form to achieve down-weighting, (e.g. the Gaussian kernel $K(u) = (2\pi)^{-1/2}e^{-u^2/2}$, Silverman kernel $K(u) = 0.5e^{-|u|/\sqrt{2}} \cdot \sin\left(\frac{|u|}{\sqrt{2}} + \frac{\pi}{4}\right)$ for sine function sin, base of natural logarithm e and mathematical constant π; see Pagan and Ullah (1999) for further examples). Recall that HE does not natively support comparison $(\leq, \geq)$, which is needed for bounded support kernels, nor non-linear functions $(\exp, \sin)$, which is needed for non-bounded support kernels. Therefore, the main difficulty of KDE with HE is finding a kernel which can be evaluated given the limited number of available operations.

To resolve this issue, one could encode/encrypt the numbers bit-wise similarly to Song et al. (2019), Yoo et al. (2019). This requires as many ciphertexts as many bits there are. Because this is not efficient, I take a different approach which relies only on addition and multiplication.

3.3. HE-KDE

To alleviate the kernel issue with HE, I devise a polynomial which behaves like a kernel *locally*.

Definition 1 (Local Approximate Kernel)**.** $\tilde{K}_{d,\tau}(u) \coloneqq c_{d,\tau}\left(\gamma + \sum_{k=1}^{d} w_k(1+u^2-\tau)^k\right)$*, for shift* $\tau \in \mathbb{R}_{\geq 1}$*, even degree* $d \in \mathbb{N}$*, normalising constant* $c_{d,\tau}$*, the Euler constant* γ*, and weights* $w_k \coloneqq \frac{(-1)^{k+1}}{k} + (-1)^k\zeta(k+1) \in [-1,1]$ *where* $\zeta(x) = \sum_{j=1}^{\infty}\frac{1}{j^x}$ *is the Riemann zeta function.*

The motivation for $\tilde{K}_{d,\tau}$ is the kernel $u \mapsto \frac{\pi}{1+u^2}$ which has unbounded support. The problematic function for HE in this kernel is $x \mapsto 1/x$. Muqattash and Yahdi (2006) points out that the digamma function can be approximated by $\psi(x) \approx \ln(x) - 1/x$. Starting from here, I write $1/x \approx \ln(x) - \psi(x)$ and use the degree-d Taylor-approximations of $\ln(x)$ and $\psi(x)$ with a shift τ. [5]

What is the role of τ? How is $c_{d,\tau}$ computed? Figure 3 depicts $\tilde{K}_{d,\tau}$ for various degrees and shifts. On one hand, by increasing τ, we can increase the region where $\tilde{K}_{d,\tau}$ can perform "sensible weighting". Let $l_1(d,\tau) \coloneqq \inf\{u \in \mathbb{R}_+ : \tilde{K}_{d,\tau}(u) < \tilde{K}_{d,\tau}(u+\varepsilon),\ \varepsilon > 0\}$ be the infimum positive value where $\tilde{K}_{d,\tau}$ starts to increase. Throughout, I fix $\varepsilon \coloneqq 10^{-2}$. l_1, numerically approximated, is also depicted in Figure 3. By symmetry, the weights induced by $\tilde{K}_{d,\tau}$ are increasing in $|u|$ for $u \notin [\pm l_1(d,\tau)]$, hence when applied to KDE, observations with large $|x - x_i|$ would get large weights. This is clearly undesirable, so we need to make sure that $u \in [\pm l_1(d,\tau)]$, where $[\pm l_1(d,\tau)]$ is the "sensible weighting" region. Given l_1, $c_{d,\tau}$ is computed numerically such that $\int_{-l_1(d,\tau)}^{l_1(d,\tau)} \tilde{K}_{d,\tau}(u)du = 1$, from the unnormalised version of $\tilde{K}_{d,\tau}$. On the other hand, by increasing τ we also increase the middle, non-zero region of $\tilde{K}_{d,\tau}$, decreasing the region where $\tilde{K}_{d,\tau}$ can perform "effective *down*-weighting". This prevents the kernel to attribute small weight to a dissimilar observation when applied to KDE. Let $l_2(d,\tau) \coloneqq \inf\{u \in \mathbb{R}_+ : \tilde{K}_{d,\tau}(u) < \delta,\ \delta > 0\}$ be the infimum positive value where $\tilde{K}_{d,\tau}$ becomes near-zero for small δ. Throughout, I fix $\delta \coloneqq 10^{-3}$.[6] The "effective down-weighting" region is $\{u \in \mathbb{R} : |u| \geq l_2(d,\tau)\}$. l_2, numerically approximated, is also depicted in Figure 3. To compare the trade-off between the "sensible weighting" and "effective down-weighting" region, the dependence of l_1, l_2 on τ is illustrated in Figure 4. Remarkably, we have $l_1(d,\tau) \approx \sqrt{\tau}$ from the first column, so that the "sensible weighting" region is $O(\sqrt{\tau})$. Moreover, l_2 increases slower than $\sqrt{\tau}$ in τ for all degree d, so that the the "effective down-weighting" region decreases as $o(\sqrt{\tau})$.

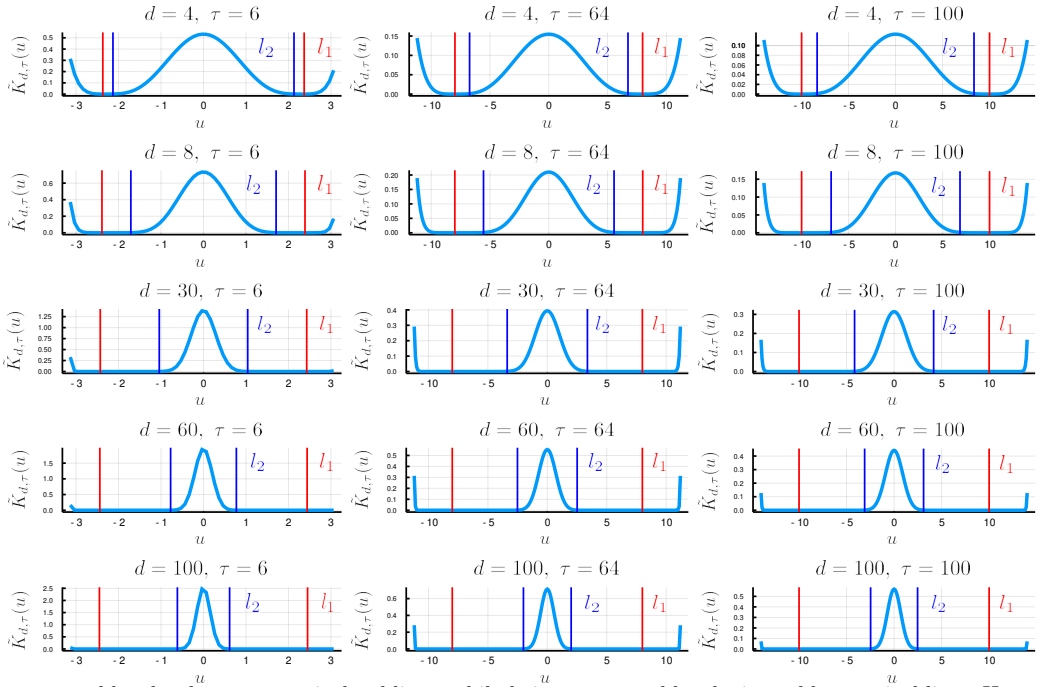
What is the role of d? Figures 3 and 4 illustrate the effect of d on l_1, l_2. While l_1 does not change with d, l_2 can be decreased by increasing d, irrespective of τ. That is, the "effective down-weighting" region can be increased by increasing d. As a consequence, by increasing τ or

[5] Of course, I could use the Taylor-approximation of $x \mapsto 1/x$ right away, which is $x \mapsto \sum_{j=0}^{\infty}(-1)^k(x-1)^k$ around 1. In fact, w_k oscillates in [-1, 1], similarly to $(-1)^k$. Numerical analysis suggests, however, that $\tilde{K}_{d,\tau}$ has slightly better properties: the minimum of the kernel tends to be closer to zero for the digamma approach.

[6] If the Taylor-approximation of $x \mapsto 1/x$ is used directly, δ needs to be higher to keep l_2 unchanged.

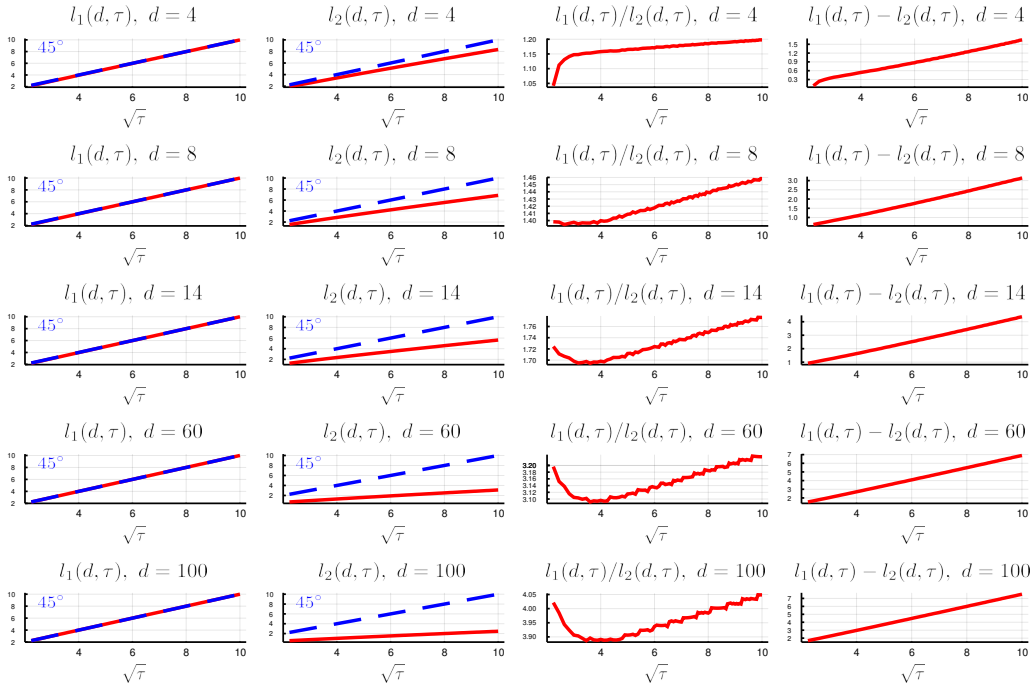
d we can improve the properties of the kernel as reflected by the third and fourth columns of Figure 4.

Figure 3: Local Approximate Kernel



Notes: $l_1$is represented by the the outer vertical red lines, while l_2 is represented by the inner blue vertical lines. Hence the "sensible weighting" region is between the red lines, and the "effective down-weighting" region is outside the blue lines.

Figure 4: Effect of Parameters on Local Approximate Kernel Properties



Having constructed the HE-compatible kernel, we can turn to the HE-compatible kernel density estimator, HE-KDE. Importantly, the bandwidth h must ensure that the input to the kernel $\frac{x-x_i}{h} \in [\pm l_1(d,\tau)]$, where $l_1(d,\tau) \approx \sqrt{\tau}$, to provide sensible weighting. If $h \geq \frac{b-a}{\sqrt{\tau}}$, then $\frac{x-x_i}{h} \in [\pm\sqrt{\tau}]$ when $x, x_i \in S = [a,b]$. As the support of f is S, $x_i \in S$ for all $i \in [n]$.

Definition 2 (HE-KDE)**.** $\tilde{f}_{d,\tau}(x) \coloneqq \frac{1}{nh}\sum_{i\in[n]} \tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)$ *for i.i.d. sample* $\boldsymbol{x} = (x_i)_{i\in[n]} \in S^n$ *from the density* f *with support* $S = [a,b]$*, query point* $x \in S$*, bandwidth* $h \geq \frac{b-a}{\sqrt{\tau}}$*.*

Having constructed the HE-compatible estimator $\tilde{f}_{d,\tau}$ on non-encrypted data, I discuss its implementation on encrypted data using the CKKS scheme. Three algorithms are designed, corresponding to the three steps in Figure 1. First, the client chooses encryption and HE-KDE parameters, then encrypts the sample $\boldsymbol{x}$ and the query point x, and sends them to the server with additional information needed for computation (Algorithm 1). Second, the cloud computes $\tilde{f}_{d,\tau}(x)$ homomorphically on the encrypted data and sends the encrypted result to the client (Algorithm 2). Third, the client decrypts the result and hence obtains $\tilde{f}_{d,\tau}(x)$ (Algorithm 3).

Note that the encrypted data are $\boldsymbol{x}$ and x; none of $n, h, d, \tau, c_{d,\tau}, (w_k)_{k\in[d]}$ are encrypted. Because τ enters $\tilde{f}_{d,\tau}$ only linearly, except $c_{d,\tau}$, τ may be encrypted if one designs an encrypted look-up algorithm to access $c_{d,\tau}$, similarly to Zhang et al. (2015). Encrypting τ could prevent the server to infer properties of the data from h – since the optimal bandwidth h, presented in the next section, might depend on data properties.

Algorithm 1 HE-KDE (CKKS) Step 1: Client encrypts data

input: sample $\boldsymbol{x} = (x_i)_{i\in[n]} \in S^n$, $n = N/2 \geq H/2$ for power-of-2 N, encryption parameter H of $\mathcal{HWT}(H)$; query point $x \in S$; minimum security requirement λ_0

output: encrypted data, evaluation and switching keys, HE-KDE parameters

1: set degree d and shift τ based on n ▷ HE-KDE parameters
2: set bandwidth h based on d, τ, n, S
3: set scale Δ based on required precision and S ▷ encryption parameters
4: set initial level L based on $d, \tau, \Delta, \lambda_0$
5: generate keys $(\mathsf{sk}, \mathsf{pk}, \mathsf{evk}) \coloneqq \mathsf{KeyGen}()$; keep sk in secret
6: generate set of switching keys $(\mathsf{swk}_{i,j})_{i,j\in T}$ where $\mathsf{swk}_{i,j} \coloneqq \mathsf{KSGen}(\kappa_k(\mathsf{sk}))$ with $k = j^{-1}i \pmod{2N}$ for all $i, j \in T$
7: $\boldsymbol{c}_{\text{sample}} \coloneqq \mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(\boldsymbol{x}))$ ▷ encrypt data
8: $\boldsymbol{c}_{\text{query}} \coloneqq \mathsf{Enc}_{\mathsf{pk}}(\mathsf{Ecd}_\Delta(-x\mathbf{1}_n))$ ▷ encrypt $(-\mathbf{1}_n)$ times query point
9: **return** $(\boldsymbol{c}_{\text{sample}}, \boldsymbol{c}_{\text{query}}, \mathsf{evk}, (\mathsf{swk}_{i,j})_{i,j\in T}, \Delta, n, h, d, \tau, c_{d,\tau})$

Algorithm 2 HE-KDE (CKKS) Step 2: Cloud computes estimator

input: output of Algorithm 1

output: encryption of HE-KDE estimator $\tilde{f}_{d,\tau}(x)$

$\check{\tau} \coloneqq \mathsf{Ecd}_\Delta((1-\tau)\mathbf{1}_n)$ ▷ encode (transformed) parameters

$\check{\gamma} \coloneqq \mathsf{Ecd}_\Delta((nh)^{-1}c_{d,\tau}\gamma\mathbf{1}_n)$

$\check{w}_k \coloneqq \mathsf{Ecd}_\Delta((nh)^{-1}c_{d,\tau}w_k\mathbf{1}_n)$ for $k \in [d]$

$\check{h} \coloneqq \mathsf{Ecd}_\Delta(h^{-1}\mathbf{1}_n)$

$\boldsymbol{c} \coloneqq \mathsf{RS}(\mathsf{CMult}(\mathsf{Add}(\boldsymbol{c}_{\text{sample}}, \boldsymbol{c}_{\text{query}}), \check{h}))$ ▷ input to $\tilde{K}_{d,\tau}$: $\frac{x\mathbf{1}_n - \boldsymbol{x}}{h} =: \boldsymbol{u}$

$\boldsymbol{c} \coloneqq \mathsf{RS}(\mathsf{Mult}_{\mathsf{evk}}(\boldsymbol{c}, \boldsymbol{c}))$ ▷ computing $\tilde{K}_{d,\tau}$: $\boldsymbol{u}^2$

$\boldsymbol{c} \coloneqq \mathsf{CAdd}(\boldsymbol{c}, \check{\tau})$ ▷ computing $\tilde{K}_{d,\tau}$: $\mathbf{1}_n - \tau\mathbf{1}_n + \boldsymbol{u}^2 =: \boldsymbol{y}$

$\boldsymbol{c}_{\text{power}} \coloneqq \boldsymbol{c}$

$\boldsymbol{c}_{\text{term}} \coloneqq \mathsf{RS}(\mathsf{CMult}(\boldsymbol{c}_{\text{power}}, \check{w}_1))$ ▷ computing $(nh)^{-1}\tilde{K}_{d,\tau}$: $(nh)^{-1}c_{d,\tau}w_1\boldsymbol{y}^1$

$\boldsymbol{c}_{\text{result}} \coloneqq \boldsymbol{c}_{\text{term}}$

for $k \in \{2, 3, \ldots, d\}$ **do**

  $\boldsymbol{c}_{\text{power}} \coloneqq \mathsf{RS}(\mathsf{Mult}_{\mathsf{evk}}(\boldsymbol{c}_{\text{power}}, \boldsymbol{c}))$ ▷ computing $(nh)^{-1}\tilde{K}_{d,\tau}$: $\boldsymbol{y}^k$

  $\boldsymbol{c}_{\text{term}} \coloneqq \mathsf{RS}(\mathsf{CMult}(\boldsymbol{c}_{\text{power}}, \check{w}_k))$ ▷ computing $(nh)^{-1}\tilde{K}_{d,\tau}$: $(nh)^{-1}c_{d,\tau}w_k\boldsymbol{y}^k$

  $\boldsymbol{c}_{\text{result}} \coloneqq \mathsf{ModDown}(\boldsymbol{c}_{\text{result}}, \text{modulus of } \boldsymbol{c}_{\text{term}})$ ▷ decrease modulus for addition:

  $\boldsymbol{c}_{\text{result}} \coloneqq \mathsf{Add}(\boldsymbol{c}_{\text{result}}, \boldsymbol{c}_{\text{term}})$ ▷ computing $(nh)^{-1}\tilde{K}_{d,\tau}$: $\sum_{j=1}^k (nh)^{-1}c_{d,\tau}w_j\boldsymbol{y}^j$

$\boldsymbol{c}_{\text{result}} \coloneqq \mathsf{CAdd}(\boldsymbol{c}_{\text{result}}, \check{\gamma})$ ▷ computing $(nh)^{-1}\tilde{K}_{d,\tau}$: $(nh)^{-1}c_{d,\tau}\left(\gamma + \sum_{j=1}^k w_j\boldsymbol{y}^j\right) = \left((nh)^{-1}\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)\right)_{i\in[n]}$

$\boldsymbol{c}_{\text{result}} \coloneqq \mathsf{Sum}_{(\mathsf{swk}_{i,j})_{i,j\in T}}(\boldsymbol{c}_{\text{result}})$ ▷ computing $\tilde{f}_{d,\tau}$: $(nh)^{-1}\sum_{i\in[n]}\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)$

return $\boldsymbol{c}_{\text{result}}$

Algorithm 3 HE-KDE (CKKS) Step 3: Client decrypts estimator

input: output of Algorithm 2; secret key sk

output: HE-KDE estimator $\tilde{f}_{d,\tau}(x)$

$\boldsymbol{z} \coloneqq \mathsf{Dcd}_\Delta(\mathsf{Dec}_{\mathsf{sk}}(\boldsymbol{c}_{\text{result}})) \in \mathbb{C}^n$ ▷ decrypt then decode result

$\boldsymbol{z}_r \coloneqq \Re(\boldsymbol{z}) \in \mathbb{R}^n$, where $\Re(\boldsymbol{z})$ are the real parts of $\boldsymbol{z}$ ▷ remove imaginary noise due to encryption

set $\tilde{f}_{d,\tau}(x)$ to first element of $\boldsymbol{z}_r$ ▷ other elements are noise produced by Sum

return $\tilde{f}_{d,\tau}(x)$

Proposition 1 (Multiplicative Depth of Algorithm 2)**.** *The multiplicative depth ($D_{\mathsf{(C)Mult}}$) of Algorithm 2 is $d+2$, where d is the kernel degree in Definition 1.*

Proof. Appendix A.1.

3.4. HE-KDE Properties

In this section, I analyse the properties of HE-KDE on non-encrypted data. Since a function $f : \mathbb{R} \to \mathbb{R}$ is a density if and only if $f(x) \geq 0$ for all $x \in \mathbb{R}$ and $\int_{\mathbb{R}} f(x)dx = 1$, it is desirable

that a density estimator exhibits the same properties. The following proposition states that HE-KDE is always non-negative, but it need not integrate to one on S, hence it potentially induces a defective cumulative distribution function with $\lim_{x\to b}\int_a^x \tilde{f}_{d,\tau}(t)dt \leq 1$.

Proposition 2 (Bounds)**.** *For all $\tau \in \mathbb{R}_{\geq 1}$, for all even $d \in \mathbb{N}$, for all $S = [a, b] \subset \mathbb{R}$, we have*

(i) $\tilde{f}_{d,\tau}(x) \geq 0$ *for all* $x \in S$*, and*

(ii) $\int_S \tilde{f}_{d,\tau}(x)dx \leq 1$.

Proof. Appendix A.2.

Pointwise variance and bias, and the Mean Integrated Square Error (MISE) are usual properties of interest for KDE. $\text{MISE}(\hat{f})$ for an estimator $\hat{f}$ of density f is defined as $\text{MISE}(\hat{f}) \coloneqq \int_{\mathbb{R}} \mathbb{E}_f\left[(\hat{f}(x) - f(x))^2\right] dx = \int_{\mathbb{R}} \left(\mathbb{V}_f\left[\hat{f}(x)\right] + \mathbb{B}\text{ias}_f(\hat{f}(x))^2\right) dx$, for $\mathbb{B}\text{ias}_f(\hat{f}(x)) \coloneqq \mathbb{E}_f\left[\hat{f}(x)\right] - f(x)$. The smaller is MISE, the closer is the estimator $\hat{f}$ to the estimand f. Proposition 3 asserts that these properties are similar to that of KDE on non-encrypted data, using usual kernels. The integrals I_1, I_2 of the variance and bias, plotted in Figure 11, play important roles in the rest of this section.

Proposition 3 (MISE)**.** *Assume that f is twice continuously differentiable on $\mathbb{R}$, and that $\tau \in \mathbb{R}_{\geq 1}$, and even number $d \in \mathbb{N}$ are such that $\tilde{K}_{d,\tau}(z)^2 < \infty\ \forall z \in [\pm\sqrt{\tau}]$. Then the pointwise variance and bias, with respect to f, for all $x \in S$ satisfy*

(i) $\mathbb{V}_f\left[\tilde{f}_{d,\tau}(x)\right] \leq V(x, n, h, d, \tau)$ *where* $V(x, n, h, d, \tau) \approx \frac{f(x)}{nh} I_1(d, \tau)$ *for n large and h small, with* $I_1(d, \tau) \coloneqq \int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)^2 dz$*, and*

(ii) $\mathbb{B}\text{ias}_f(\tilde{f}_{d,\tau}(x)) = \frac{h^2 f''(x)}{2} I_2(d, \tau) + o(h^2)$ *for h small, where* $I_2(d, \tau) \coloneqq \int_{-\sqrt{\tau}}^{\sqrt{\tau}} z^2 \tilde{K}_{d,\tau}(z)dz$.

Further assume that there exists $\beta_f < \infty$ such that $\int_S f''(x-\beta)^2 dx \leq \beta_f$ for all $\beta \in \mathbb{R}$ uniformly. Then the mean integrated square error (MISE) satisfies

(iii) $\text{MISE}(\tilde{f}_{d,\tau}) \leq M_{d,\tau}(h) \coloneqq \frac{h_1(d,\tau)}{nh} + h^4 h_2(d, \tau)$ *where* $h_1(d, \tau) \coloneqq I_1(d, \tau)$ *and* $h_2(d, \tau) \coloneqq I_2(d, \tau)^2 \frac{\beta_f}{3}$.

Proof. Appendix A.3.

In non-encrypted KDE, the bandwidth h is chosen to minimise the MISE bound, $M_{d,\tau}$, achieved by setting h proportional to $n^{-1/5}$, which balances the variance and bias terms in MISE (Pagan and Ullah, 1999). Hence $\text{MISE}(h)|_{h\propto n^{-1/5}}$ vanishes for large sample size as $\text{MISE}(h)|_{h\propto n^{-1/5}} = O\left(n^{-4/5}\right)$.

HE-KDE is different. Proposition 4 characterises the optimal bandwidth and MISE-bound for HE-KDE. The non-encrypted rate $h \propto n^{-1/5}$ need not be a feasible solution. It is because h is subject to $h \geq \frac{b-a}{\sqrt{\tau}}$ to make sure that the input to the HE kernel $\tilde{K}_{d,\tau}$, $\frac{x-x_i}{h}$, falls in the "sensible

weighting" region $[\pm\sqrt{\tau}]$ for all $x \in S$, for all $i \in [n]$. For fixed d and τ, an increasing sample size n inevitably leads to a binding constraint $h = \frac{b-a}{\sqrt{\tau}} = h^*_{\text{bind}}(d,\tau)$, therefore, $M_{d,\tau}(h^*(d,\tau,n)) = M_{d,\tau}(h^*_{\text{bind}}(d,\tau))$. However, $M_{d,\tau}(h^*_{\text{bind}}(d,\tau))$ does not vanish as the sample size increases, unlike $O\left(n^{-4/5}\right)$ for non-encrypted KDE. The non-vanishing MISE-term, $\frac{(b-a)^4}{\tau^2}h_2(d,\tau)$, can be quite large for large support of f, $S = [a,b]$, and non-smooth densities (h_2 is increasing in β_f). In the next section, using numerical methods, I show that this issue can be resolved by choosing (d,τ) appropriately.

Proposition 4 (Optimal Bandwidth)**.** *Let $h^*(d,\tau,n)$ be a solution to the program*

$$\min_{h\in\mathbb{R}_{++}} M_{d,\tau}(h) \text{ subject to } h \geq \frac{b-a}{\sqrt{\tau}}.$$

Then, given a, b, n, d, τ, the solution $h^*(d,\tau,n)$ exists, is unique, and satisfies

$$h^*(d,\tau,n) = \begin{cases} h^*_{bind}(\tau) \coloneqq \frac{b-a}{\sqrt{\tau}} & \text{if} \quad \frac{b-a}{\sqrt{\tau}} \geq \left(\frac{h_1(d,\tau)}{4h_2(d,\tau)n}\right)^{1/5} \\ h^*_{nonbind}(d,\tau) \coloneqq \left(\frac{h_1(d,\tau)}{4h_2(d,\tau)n}\right)^{1/5} & \text{otherwise} \end{cases}$$

$$M_{d,\tau}(h^*_{bind}(\tau)) = \frac{\sqrt{\tau}h_1(d,\tau)}{n(b-a)} + \frac{(b-a)^4}{\tau^2}h_2(d,\tau) \geq$$

$$M_{d,\tau}(h^*_{nonbind}(d,\tau)) = \left(2^7 h_2(d,\tau)h_1(d,\tau)^4 n^{-4}\right)^{1/5}.$$

Proof. Appendix A.4

3.5. Choosing Parameters: Statistical versus Encryption Optimality

How to choose degree d and shift τ? The choice induces a trade-off between good statistical and encryption properties of HE-KDE. First, choices that lead to good statistical properties are discussed in this subsection. Second, these choices are shown to exhibit unacceptable encryption properties, and a solution balancing between statistical and encryption properties is designed.

In either case, I focus on choices that minimise the MISE-bound. Analytic solutions to minimising $M_{d,\tau}(h^*(d,\tau,n))$ with respect to (d,τ) can depend on, and only on, the sample size n, the support S, and smoothness β_f. As a heurestic method, I only consider sequences d_n, τ_n of the sample size n that ensure the convergence of MISE-bound as $n \to \infty$. Heurestics are used because analytical or numerical solutions are non-existent or intractable. I choose d_n, τ_n so that $M_{d,\tau}(h^*_{\text{bind}}(\tau))$ is minimised. This further simplifies the analysis, and is a conservative choice in that $M_{d,\tau}(h^*_{\text{bind}}(\tau)) \geq M_{d,\tau}(h^*_{\text{nonbind}}(d,\tau))$ for all choices of $\tau \in \mathbb{R}_{\geq 1}$ and even $d \in \mathbb{N}$. Thus, $M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n)) \to 0$ implies $M_{d_n,\tau_n}(h^*_{\text{nonbind}}(d_n,\tau_n)) \to 0$ as $n \to \infty$, therefore $M_{d_n,\tau_n}(h^*(d_n,\tau_n,n)) \to 0$ as $n \to \infty$.

3.5.1 Statistical Optimality

When the constraint is binding, $h^*_{\text{bind}}(\tau) = \frac{b-a}{\sqrt{\tau}}$. This suggests setting $\tau_n \coloneqq n^{2/5}$, so that $h^*_{\text{bind}}(\tau_n) \propto n^{-1/5}$ similarly to the non-encrypted KDE case. Then by Proposition 4 we have

$M_{d,\tau_n}(h^*_{\text{bind}}(\tau_n)) = \frac{h_1(d,\tau_n)}{n^{4/5}(b-a)} + \frac{(b-a)^4}{n^{4/5}} h_2(d,\tau_n)$. This suggests setting d_n so that both $n^{-4/5}h_1(d_n,\tau_n)$ and $n^{-4/5}h_2(d_n,\tau_n) \to 0$ as $n \to \infty$ for $\tau_n = n^{2/5}$. By definition, $h_1(d,\tau) = I_1(d,\tau)$ and $h_2(d,\tau) = I_2(d,\tau)^2\frac{\beta_f}{3}$. The integrals $I_1, I_2 \geq 0$ are plotted in Figure 11. $I_1(d,\tau)$ is decreasing in τ for all d, and non-increasing in d as τ increases. $I_2(d,\tau)$ is increasing in τ for all d, and decreasing in d as τ increases. Therefore, setting d_n to an increasing sequence of n is a sensible choice because it (i) offsets the increase in h_2 caused by the increasing τ_n sequence; and (ii) causes no problems for h_1, which decays to zero as τ_n increases.

How fast should d_n increase with n? Theoretically, points (i) and (ii) imply that the faster, the better. Practically, several factors restrict d_n. First, the larger is the degree, the longer it takes to evaluate $\tilde{K}_{d,\tau}$ as its time complexity is $O(d)$ conditional on having computed $c_{d,\tau}$; if $c_{d,\tau}$ needs to be computed, its complexity is increased depending on the numerical integrator. Second, computing term $(1+u^2-\tau)^d$ of $\tilde{K}_{d,\tau}$ for large d either becomes imprecise due to limited numerical precision, or, when a data type supporting arbitrary-precision arithmetic is used, results in increased evaluation time. Hence, the choice is somewhat arbitrary. I set $d_n \coloneqq \lfloor n^{4/5} \rceil_2 < n$, where $\lfloor . \rceil_2$ means rounding to the nearest even integer, and numerically analyse the MISE-bound for this choice.

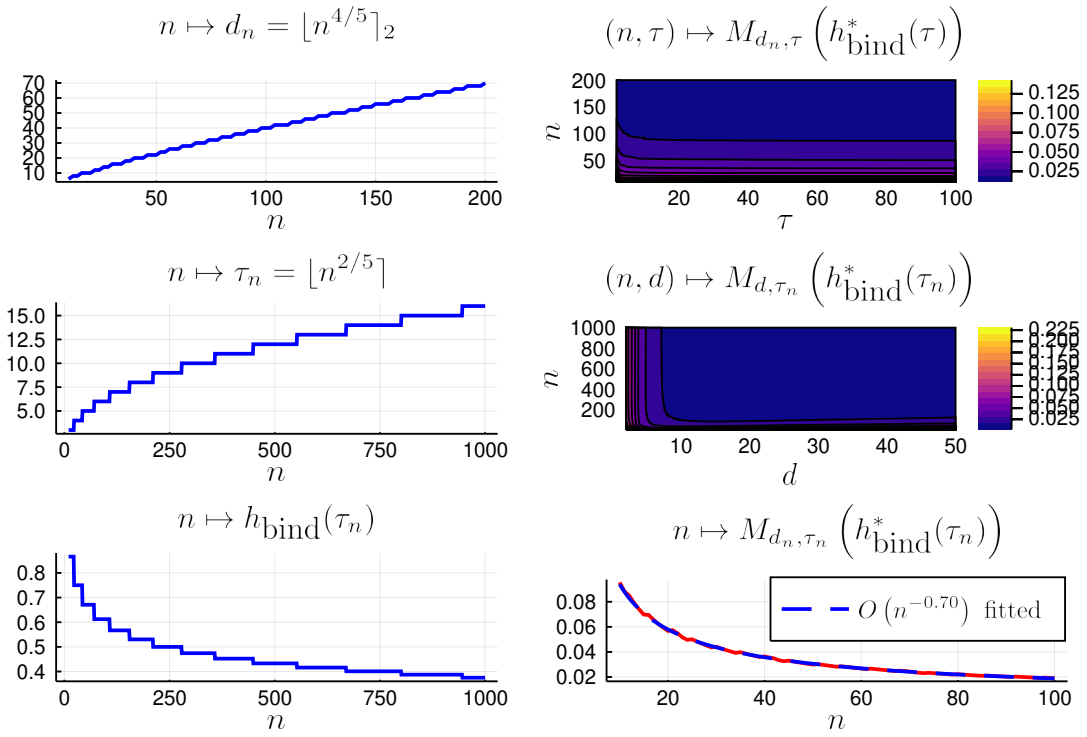
Is $d_n = \lfloor n^{4/5} \rceil_2$ and $\tau_n = n^{2/5}$ sufficient for $M_{d_n,\tau_n}(h^*(d_n,\tau_n,n)) \to 0$ as $n \to \infty$? Figure 5 depicts sequences d_n, τ_n, bandwidth $h^*_{\text{bind}}(\tau_n)$ and objective $M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n)) \geq M_{d_n,\tau_n}(h^*(d_n,\tau_n,n))$ for given S, β_f. For computational efficiency I round τ_n to the nearest integer, that is $\tau_n = \lfloor n^{2/5} \rceil$.[7] Importantly, numerical results suggest that the sequences $d_n = \lfloor n^{4/5} \rceil_2, \tau_n = \lfloor n^{2/5} \rceil$ are sufficient for $M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n)) \to 0$ as $n \to \infty$, which in turn implies $M_{d_n,\tau_n}(h^*(d_n,\tau_n,n)) \to 0$.

What is the impact of $S = [a,b]$ and β_f on (the convergence of) the MISE-bound? By definition, $M_{d,\tau}(h^*_{\text{bind}}(\tau)) = \frac{\sqrt{\tau}h_1(d,\tau)}{n(b-a)} + \frac{(b-a)^4}{\tau^2} h_2(d,\tau)$, where $h_1(d,\tau) = I_1(d,\tau)$ and $h_2(d,\tau) = I_2(d,\tau)^2\frac{\beta_f}{3}$. A larger S is associated with larger $b-a$. Hence a larger support has the same effect as an increasing β_f. Namely, larger S (or β_f) increases the influence of the second term in the objective, $\frac{(b-a)^4}{\tau^2} h_2(d,\tau)$, relatively to the first one, $\frac{\sqrt{\tau}h_1(d,\tau)}{n(b-a)}$. As a consequence , the convergence becomes more dominated by the second, bias term than before. This is analysed in more detail in Appendix C.1.1.

To sum up, setting $h \coloneqq \frac{b-a}{\sqrt{\tau_n}}, d_n \coloneqq \lfloor n^{4/5} \rceil_2, \tau_n \coloneqq \lfloor n^{2/5} \rceil$ implies that the MISE-bound in Proposition 3, $M_{d_n,\tau_n}(h^*(d_n,\tau_n,n))$, converges to zero as the sample size n increases. The magnitude of $M_{d_n,\tau_n}(h^*(d_n,\tau_n,n))$ is affected by the support S and β_f, and so is the speed of convergence. Numerical analysis suggests, however, that $M_{d_n,\tau_n}(h^*(d_n,\tau_n,n)) \to 0$ as $n \to \infty$ for all S (and β_f) discussed here and in Appendix C.1.1.

[7] The efficiency does not come from rounding *per se*. Rather, I need to access $c_{d,\tau}$ many times during the thesis, so that I precomputed $(c_{d,\tau})_{d \in \mathbb{N}_{\leq 200}, \tau \in \mathbb{N}_{\leq 10^4}}$.

Figure 5: Statistically Optimal Parameter Choices; $S = [0, 1.5]$, $\beta_f = 3$



Notes: fitted line obtained by estimating the model $y_n = \alpha_0 n^{\alpha_1} \varepsilon_n$ with ordinary least squares after log - log transform for $y_n \coloneqq M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n))$, some error term ε_n, and model parameters α_0, α_1.

3.5.2 Encryption Optimality

While $h = \frac{b-a}{\sqrt{\tau_n}}, d_n = \lfloor n^{4/5} \rceil_2, \tau_n = \lfloor n^{2/5} \rceil$ imply the convergence of the MISE-bound, they lead to unacceptable encryption security λ. For number of observations $n = N/2$, the security estimate (1) reads $\frac{7.2n}{L} - 110 \geq \lambda$ with $L \geq (D_{\mathsf{(C)Mult}} + 1) \log_2 \Delta$. By Proposition 1, $D_{\mathsf{(C)Mult}} \propto d$. Hence $d_n \coloneqq \lfloor n^{4/5} \rceil_2$ would roughly lead to $\lambda \leq \frac{7.2n^{1/5}}{\log_2 \Delta} - 110$. Even for a small precision $\log_2 \Delta = 8$, this is non-negative only for $n \geq 2^{34}$, and reaches a 60-bit security for $n \geq 2^{38}$. HE applications surveyed in this thesis rarely use $N \geq 2^{16}$ – presumably due to slow performance.

Therefore, security considerations impose an upper bound on d_n, which in turn affects τ_n as follows. Previously, the increasing $d_n = \lfloor n^{4/5} \rceil_2$ could offset the increase in the second, bias term of $M_{d,\tau_n}(h^*_{\text{bind}}(\tau_n)) = \frac{h_1(d,\tau_n)}{n^{4/5}(b-a)} + \frac{(b-a)^4}{n^{4/5}} h_2(d, \tau_n)$ caused by the increasing $\tau_n = \lfloor n^{2/5} \rceil$. Bounded by security requirements, d_n may not be able to offset the increase. In fact, Figure 6a depicts $M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n))$ for $\tau_n = \lfloor n^{2/5} \rceil$ and constant $d_n = 100$. Instead of vanishing, the MISE-bound increases in n. Hence $\tau_n = \lfloor n^{2/5} \rceil$ becomes infeasible, especially for large S.

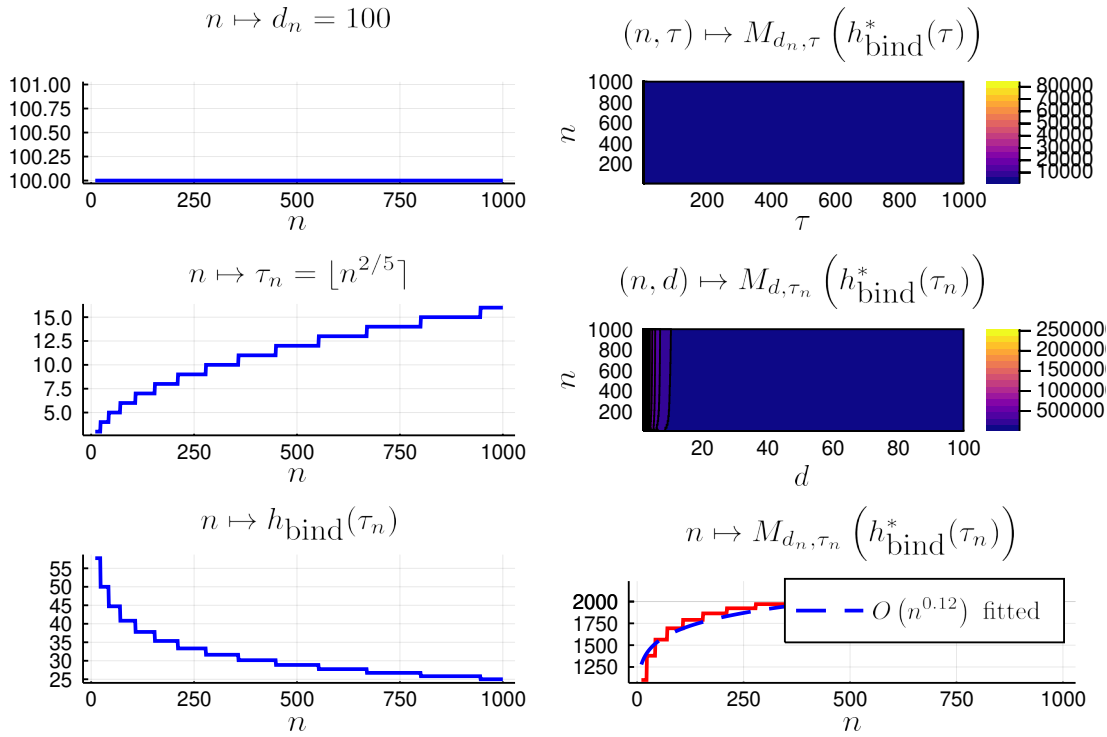
As a solution, I set d_n to a constant, $d_n \coloneqq 14$, and $\tau_n \coloneqq \lfloor n^{1/5} \rceil \leq \lfloor n^{2/5} \rceil$. In contrast to Figure 6a, $M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n)) \to 0$ as $n \to \infty$ for these sequences (see Figure 6b). Appendix C.1.2 shows that this finding is robust to even larger support up to $S = [0, 10^6]$. Regarding security, the sequence $d_n \coloneqq 14$, together with the security estimate (1) and Proposition 1

implies $\lambda \leq \frac{7.2n}{(14+2+1)\log_2 \Delta} - 110$, which is already non-negative for $n \geq 2^{12}$, given precision $\log_2 \Delta = 8$, and reaches a quite safe 6829-bit security for $n = 2^{17}$.

Last, note that if τ_n is not encrypted (it is not in this thesis) and $h = \frac{b-a}{\sqrt{\tau_n}}$, then the server can easily infer the length of $S = [a, b]$, $b - a$, from the non-encrypted h, as $b - a = \sqrt{\tau}h$. Therefore in practice, unless one encrypts τ and adjusts Algorithms 1, 2, [8] the client should send a set of bandwidths to the server, who can thus only make guesses which is the real one.

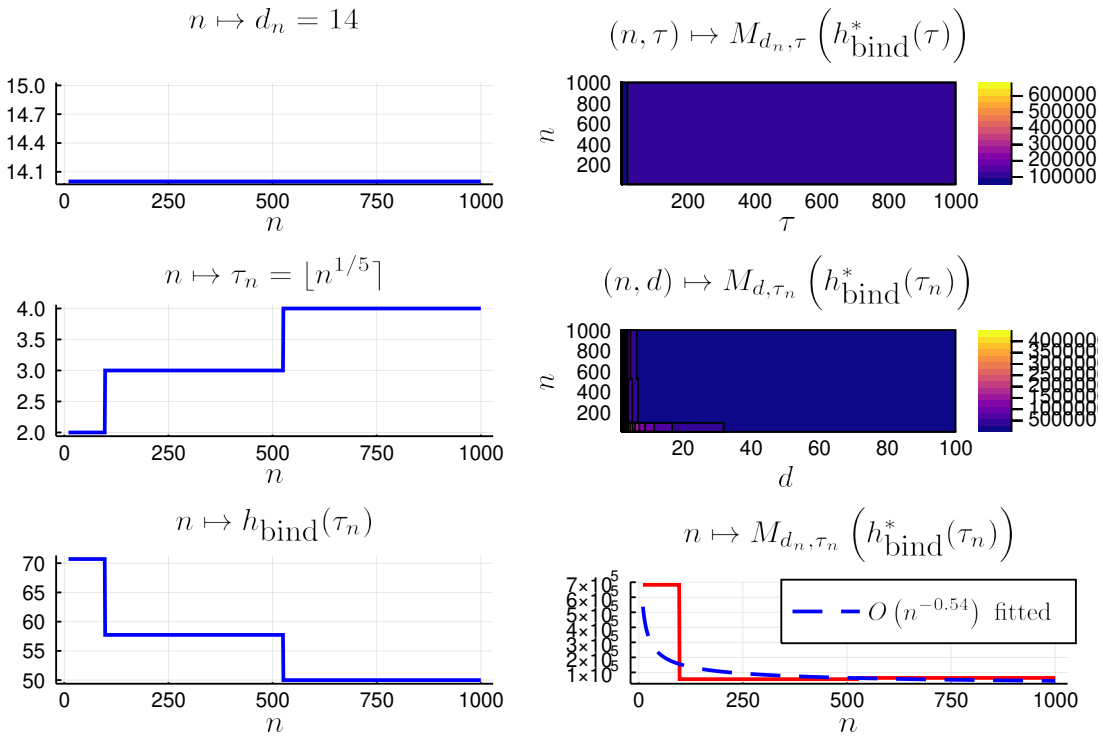
Figure 6: Encryption-optimal Parameter Choices; $S = [0, 100]$, $\beta_f = 3$

(a) Infeasible Choice



[8] E.g. by designing an encrypted look-up table for the normalising constant $c_{d,\tau}$ of $\tilde{K}_{d,\tau}$ similarly to Zhang et al. (2015). Alternatively, the client simply computes $c_{d,\tau}$, encrypts it and send it to the server. As $c_{d,\tau}$ enters $\tilde{K}_{d,\tau}$ multiplicatively and τ additively, this is feasible via the Mult and Add routines. The latter approach would increase the multiplicative depth $D_{\text{(C)Mult}}$ of Proposition 1 by 1 to $d + 3$.

(b) Feasible Choice



Notes: fitted line obtained by estimating the model $y_n = \alpha_0 n^{\alpha_1} \varepsilon_n$ with ordinary least squares after log - log transform for $y_n \coloneqq M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n))$, some error term ε_n, and model parameters α_0, α_1.

4. Examples

This section demonstrates how HE-KDE works. I begin with *non-encrypted* data in Subsection 4.1, computing $\tilde{f}_{d,\tau}$ in the clean without using Algorithms 1, 2, 3. This gives an "upper bound" on the performance of $\tilde{f}_{d,\tau}$ on encrypted data using Algorithms 1, 2, 3, which is worse due to encryption. Performance on encrypted data is addressed in Subsection 4.2.

4.1. Non-encrypted Data

I draw *i.i.d.* samples from various distributions with bounded support, and estimate the density value $f(x)$ with $\tilde{f}_{d_n,\tau_n}(x)$ *without* encrypting the sample or x. I use the statistically optimal parameter choices of Sub-subsection 3.5.1, with bandwidth $h = \frac{b-a}{\sqrt{\tau_n}}$, shift $\tau_n = \lfloor n^{2/5} \rceil$ and degree $d_n = \lfloor n^{4/5} \rceil_2$ for sample size n.

Figures 7a, 7b, 7c, 7d depict estimation for distributions Beta, Cosine, Generalised Pareto and von Mises, respectively. Each figure features different parameter values for the given distribution. Beside estimates based on a single sample, Monte Carlo (MC) simulation results are presented with a small number repetitions. In each repetition a new n-large sample is drawn, $x \mapsto \tilde{f}_{d,\tau}(x)$ is computed and plotted. Figure 14 in Appendix C.2.1 contains the same plots for other continuous, bounded-support distributions. $\tilde{f}_{d,\tau}$ is a density function estimator, not a probability mass function estimator. Thus $\tilde{f}_{d,\tau}$ is not meant for discrete distributions. As an

experiment, however, I include two discrete distributions with bounded-support in Figure 14 of Appendix C.2.1.

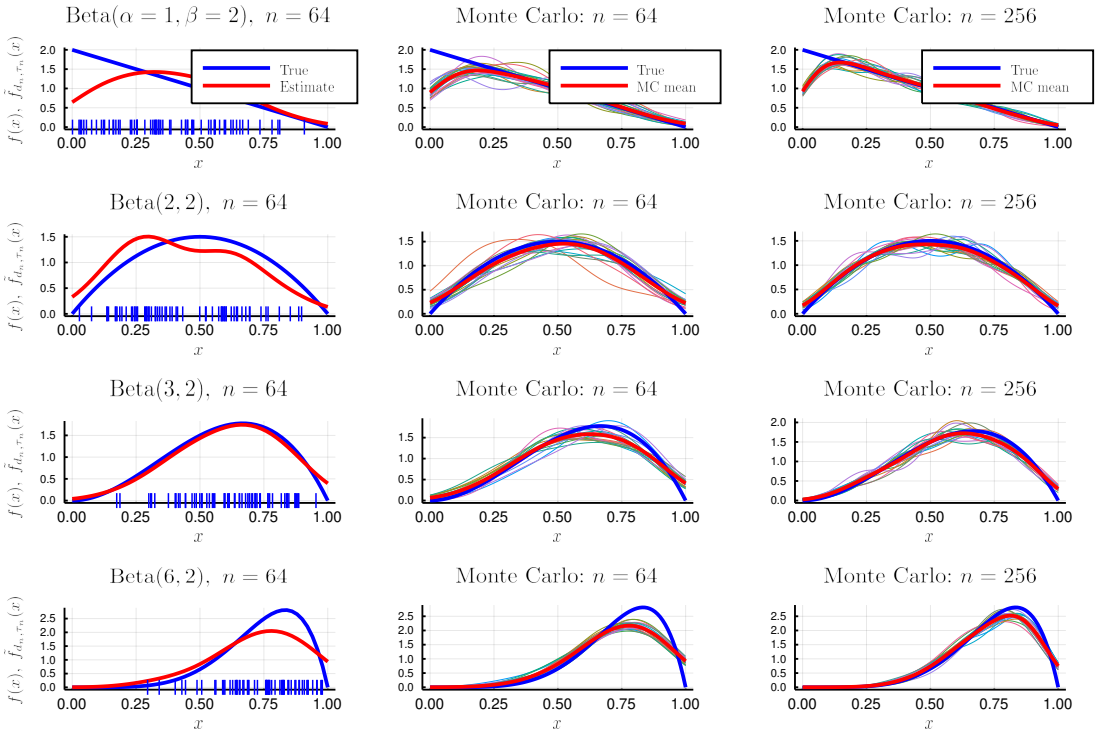
We can make four general observations from the figures about all continuous distributions above. First, HE-KDE exhibits a large bias and small variance. Second, the bias is more severe around the boundaries (this happens to usual KDE too, e.g. Leblanc (2012)). Third, HE-KDE performs better as the sample size increases, as both the (boundary) bias and variance decrease. The third point follows directly from results in the previous sections for *continuously differentiable* densities. However, the densities in most figures are not continuously differentiable, moreover, some of them are not even continuous on the boundary of their support ($\lim_{x\uparrow c} f(x) \neq \lim_{x\downarrow c} f(x)$ for $c \in \{a, b\}$). In fact, the only continuously differentiable (on $\mathbb{R}$) densities from above are that of the Cosine and the von Mises distribution (for some parameter values). Therefore, the convergent MISE (as $n \to \infty$) appears to be robust for density continuity-class assumptions for the distributions discussed here. Fourth, the variance of HE-KDE is influenced by f. E.g. the variance is very small for Beta(6,2), VM(0,3) and Generalised Pareto distributions where the probability mass is concentrated in a relatively small x-region.

The experiment with the discrete distributions reveals that HE-KDE can identify high-probability regions. It also holds that the pointwise bias and variance decreases as n increases in points where the true probability mass function is non-zero. Of course, it has a substantial non-vanishing bias at points where the probability mass function is zero as the low-variance estimator interpolates $\tilde{f}_{d,\tau}$-values in those points.

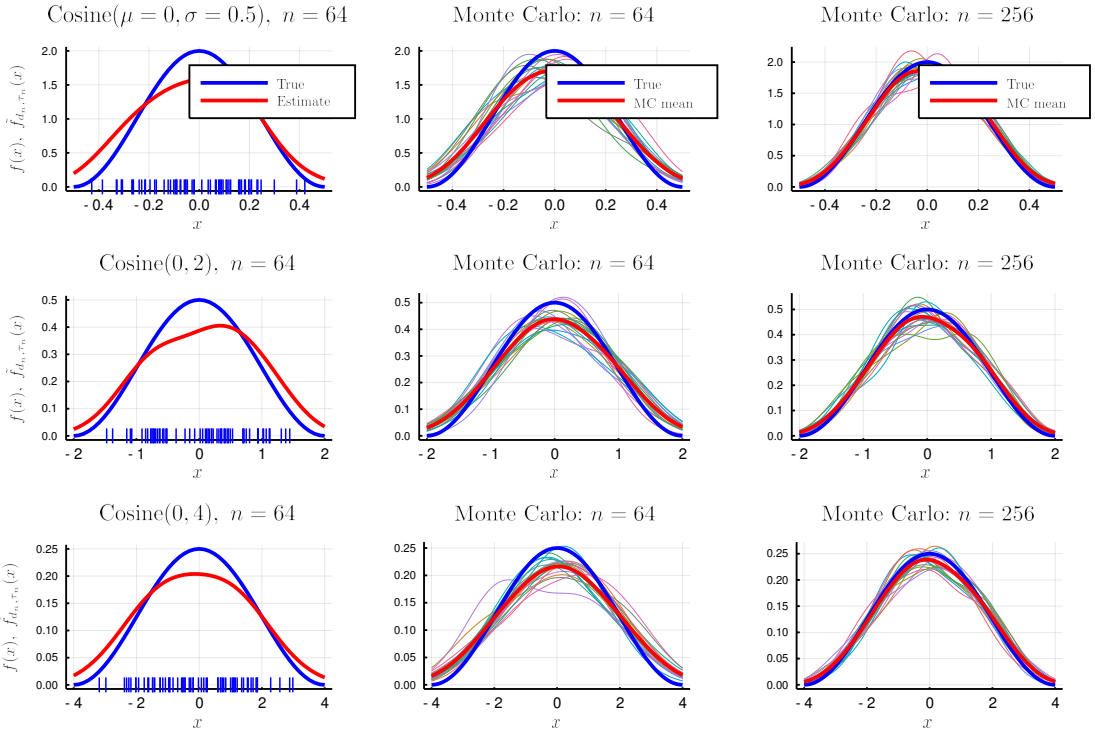
In short, HE-KDE has small variance but large bias. It performs "reasonably" for $n = 64$ and the performance improves as n increases because the variance and bias decreases. This seems to hold for all continuous distributions discussed here regardless of the continuity-class of their densities.

Figure 7: HE-KDE Non-encrypted Examples

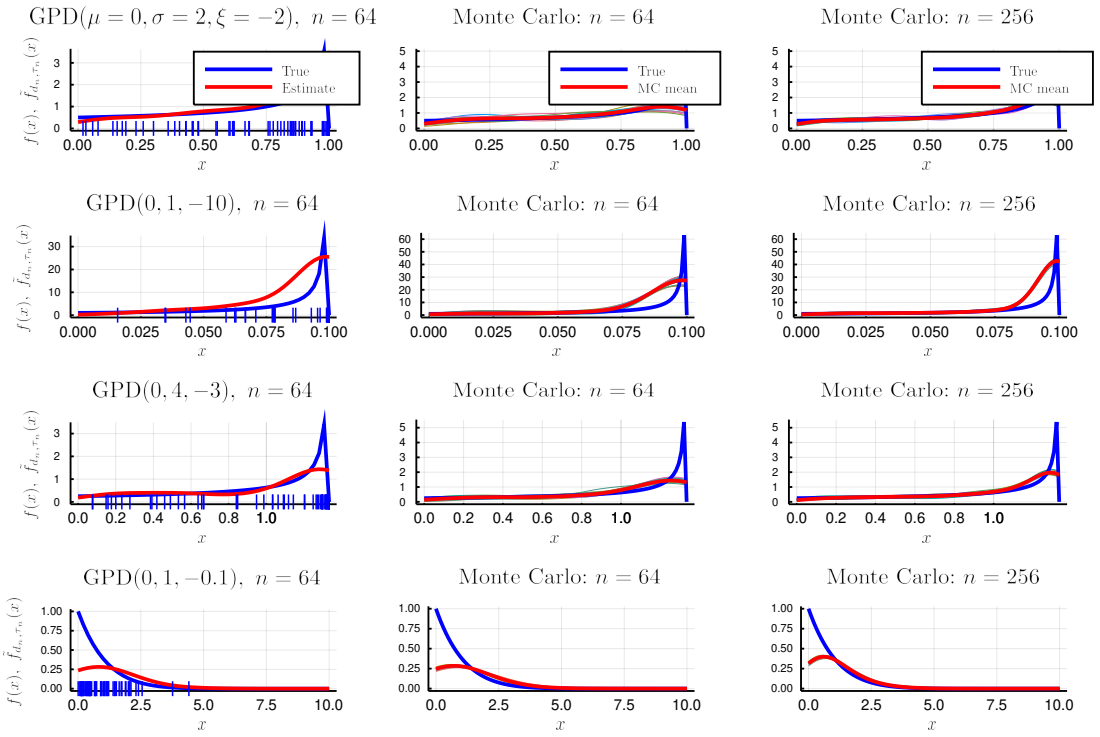
(a) Beta Distribution



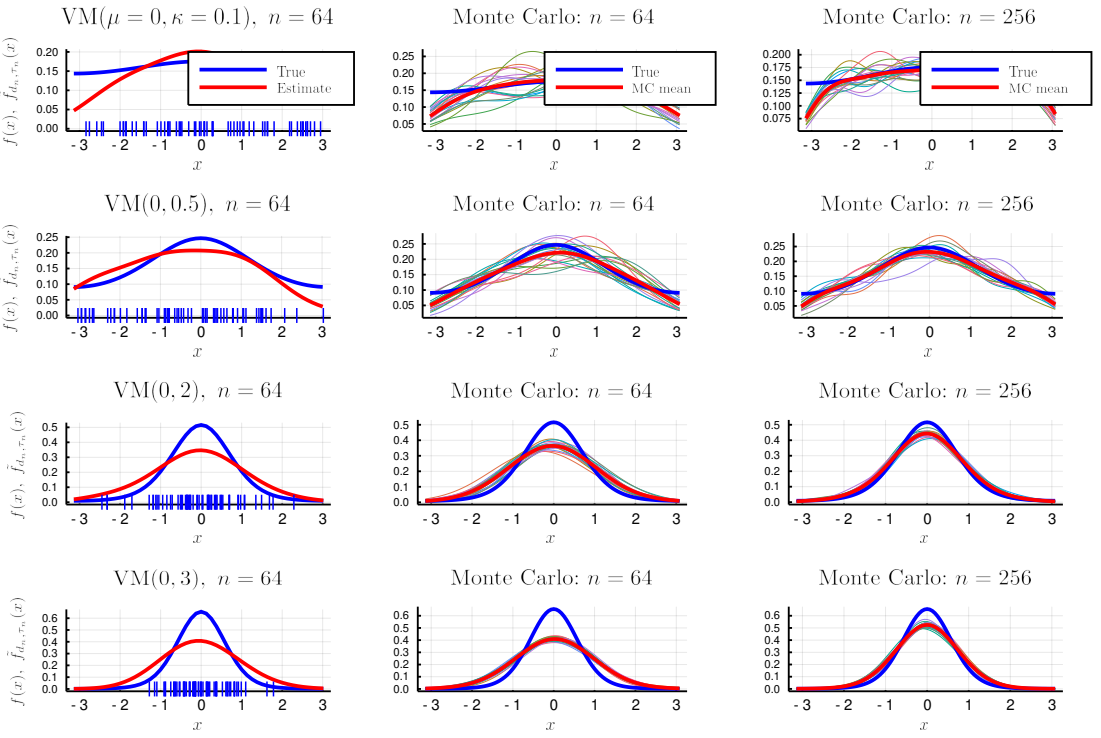
(b) Cosine Distribution



(c) Generalised Pareto Distribution



(d) von Mises Distribution



Notes: $h = \frac{b-a}{\sqrt{\tau_n}}$, $\tau_n = \lfloor n^{2/5} \rceil$, $d_n = \lfloor n^{4/5} \rceil_2$. First column: Non-encrypted HE-KDE estimate $\tilde{f}_{d_n,\tau_n}$ for a single n-large sample from true density f. Sample values indicated on the x-axis with "|". Second and third columns: Monte Carlo (MC) simulations. Each thin line, depicting non-encrypted $x \mapsto \tilde{f}_{d_n,\tau_n}(x)$, corresponds to a MC repetition. There are 20 repetitions. Thick red line is the mean of $x \mapsto \tilde{f}_{d_n,\tau_n}(x)$ across MC repetitions. Each row corresponds to different f-parameters (in brackets in the first column). f and $\tilde{f}_{d_n,\tau_n}$ are evaluated at 100 equidistant points.

4.2. Encrypted Data

The exercise in the previous subsection is repeated – this time for *encrypted* data. I draw *i.i.d.* samples from the same distributions, and estimate $f(x)$ with $\tilde{f}_{d_n,\tau_n}(x)$ using (a slightly simplified version[9] of) Algorithms 1, 2, 3 with the `HEAAN-1.0` C++ library of Cheon et al. (2017). The encryption-optimal parameter choices $h = \frac{b-a}{\sqrt{\tau_n}}, \tau_n = \lfloor n^{1/5} \rceil, d_n = 14$ are adopted from Subsubsection 3.5.2. I set the initial ciphertext level $L \coloneqq 1380$ and precision $\log_2 \Delta \coloneqq 80$, which implies a security level $\lambda \leq \frac{7.2n}{(14+2+1)\cdot 80} - 110$. Hence, for $n = 2^{15}$, $n = 2^{16}$ we have 63-bit and 236-bit security, respectively.

Similarly to the non-encrypted examples, Figures 8a, 8b, 8c, 8d depict estimation and MC results for distributions Beta, Cosine, Generalised Pareto and von Mises, respectively. The non-encrypted HE-KDE is also plotted with the encryption-optimal parameter choices. Figure 15 in Appendix C.2.2 contains the same plots for other continuous, bounded-support distributions. Two discrete distributions with bounded-support are included in Figure 15 of Appendix C.2.2. We can make the following observations about the figures.

First, computing $\tilde{f}_{d_n,\tau_n}$ on encrypted data entails no visible loss of precision compared to non-encrypted data: the graph of encrypted HE-KDE (cyan line) perfectly covers that of the non-encrypted HE-KDE (red line).

Second, all observations made about non-encrypted HE-KDE with statistically optimal parameter choices in Subsection 4.1 continue to hold. HE-KDE has small variance but large bias, especially around the boundaries of the support. Both the variance and the bias decreases, thus HE-KDE performs better, when the sample size n increases. The variance of HE-KDE is influenced by f. Observations about the discrete distribution experiment hold as well.

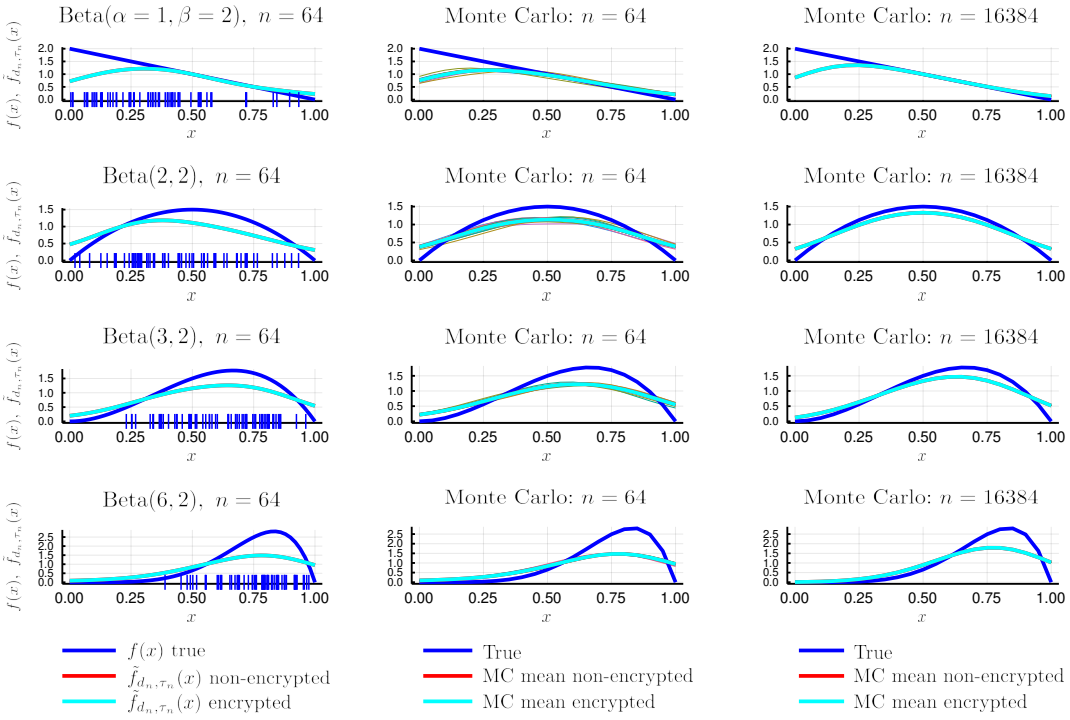
Third, there are substantial differences in the properties of HE-KDE between statistically optimal and encryption-optimal parameter choices. Compared to Figures 7a, 7b, 7c, 7d with statistically optimal parameter choices, the encryption-optimal parameter choices imply worse performance. The performance of HE-KDE with encryption-optimal parameters for $n = 2^{15} = 16384$ is worse than that of HE-KDE with statistically optimal parameters for $n = 2^6 = 64$. Recall, however, that the worse statistical performance is the price paid for security. Another difference is that encryption-optimal parameters induce smaller variance. This is a direct consequence of the first, variance term of $M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n))$ becoming smaller for smaller τ_n.

In short, HE-KDE with encryption-optimal parameters performs worse than HE-KDE with statistically optimal parameters. The bad performance is driven by the bias; the variance is very low. When the sample size increases, its performance improves and so does encryption security.

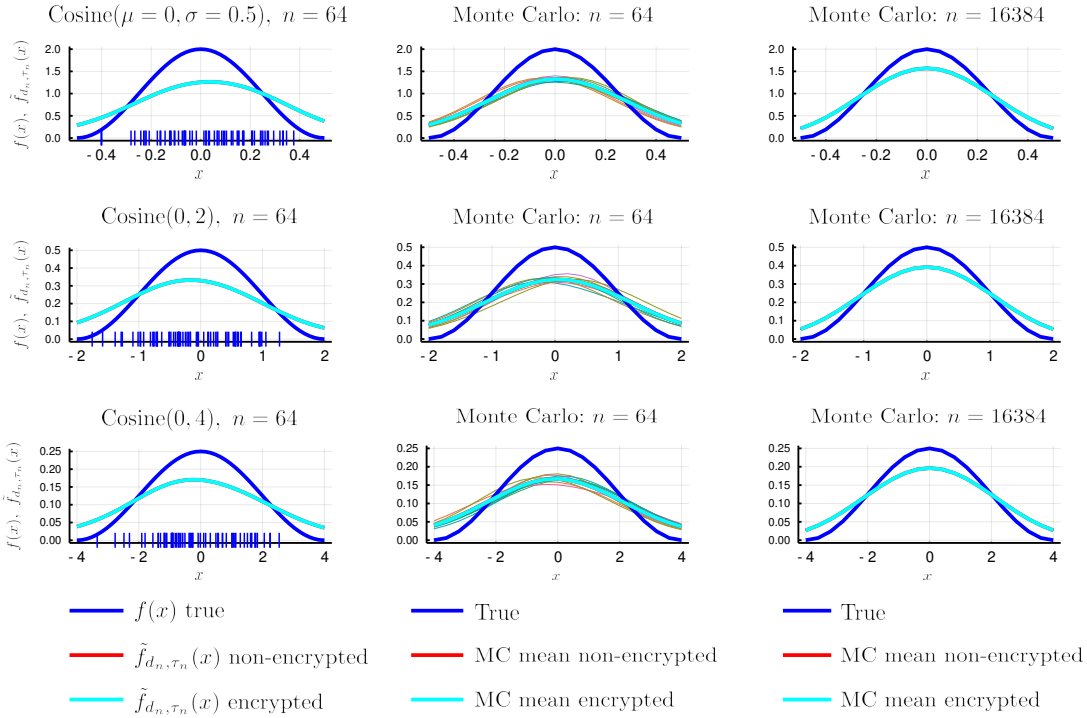
[9]The only modification is that the summation at line 17 of Algorithm 2 is performed by the client in Algorithm 3 to speed up execution.

Figure 8: HE-KDE Encrypted Examples

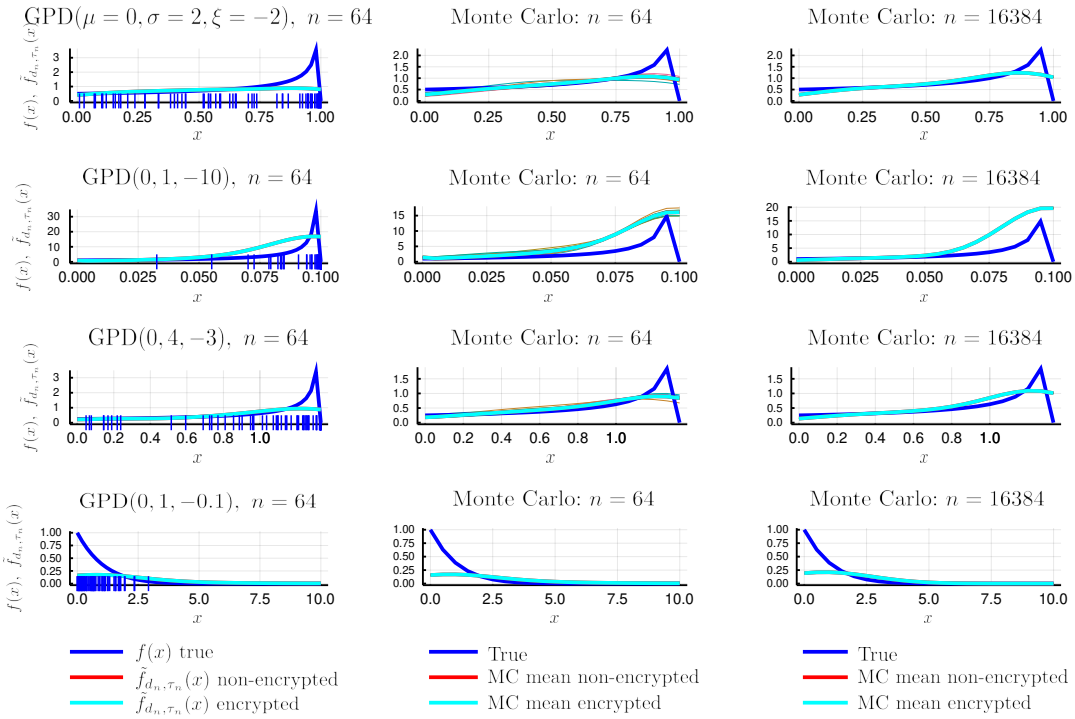
(a) Beta Distribution



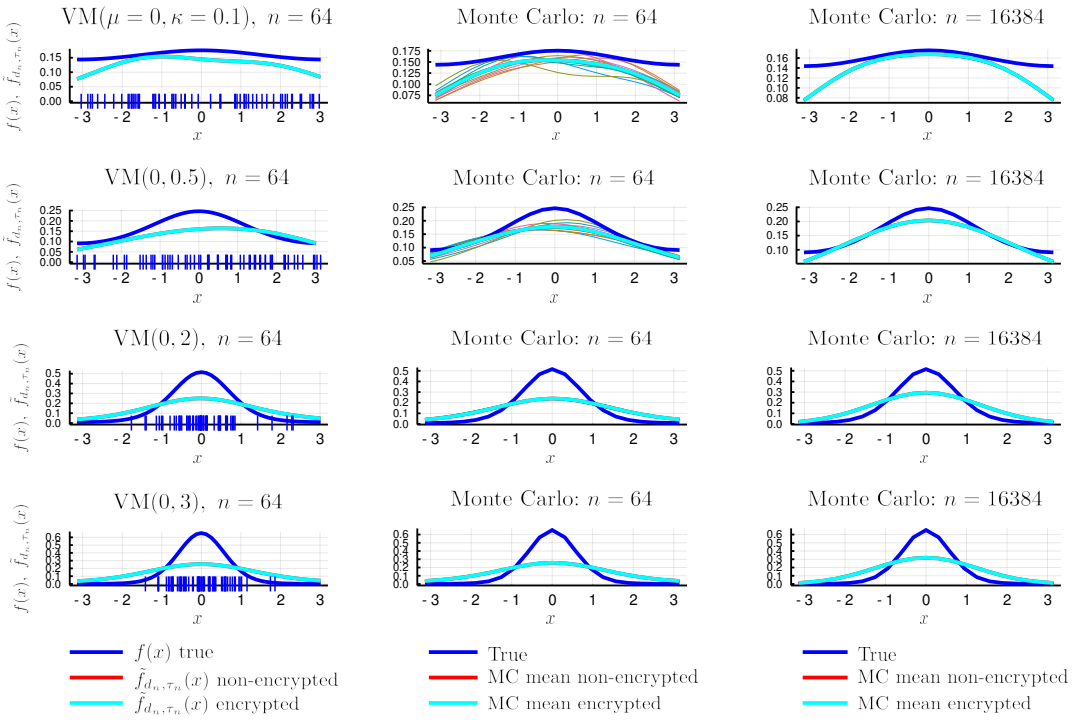
(b) Cosine Distribution



(c) Generalised Pareto Distribution



(d) von Mises Distribution



Notes: $h = \frac{b-a}{\sqrt{\tau_n}}$, $\tau_n = \lfloor n^{1/5} \rceil$, $d_n = 14$. First column: HE-KDE estimate $\tilde{f}_{d_n,\tau_n}$ (non-encrypted and encrypted) for a single n-large sample from true density f. Non-encrypted sample values indicated on the x-axis with "|". Second and third columns: Monte Carlo (MC) simulations. Each thin line, depicting $\tilde{f}_{d_n,\tau_n}$, corresponds to a MC repetition on encrypted data. Thick cyan line is the mean of $\tilde{f}_{d_n,\tau_n}(x)$ across MC repetitions on encrypted data. Thick red line is the mean of $\tilde{f}_{d_n,\tau_n}(x)$ across MC repetitions on non-encrypted data (not plotted). There are 10 repetitions. Each row corresponds to different f-parameters (in brackets in the first column). In the first column, f and $\tilde{f}_{d_n,\tau_n}$ are evaluated at 50, in the other columns, at 20 equidistant points.

4.2.1 Is the Bandwidth Too Conservative?

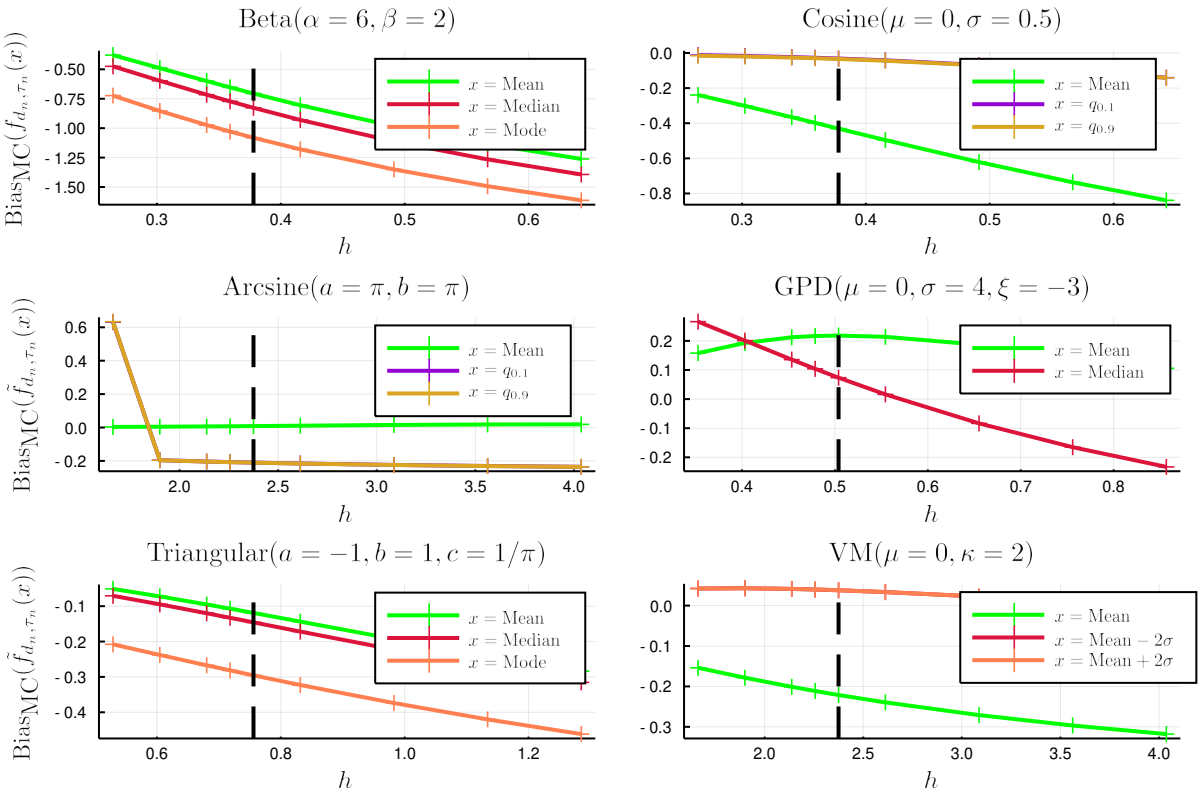
The large bias of encryption-optimal HE-KDE is due to $h = h^*_{\text{bind}}(\tau_n) = \frac{b-a}{\sqrt{\tau_n}}$ being too large as the encryption-optimal τ_n is smaller than the statistically optimal τ_n. Moreover, the choice $h \coloneqq h^*_{\text{bind}}(\tau_n)$ is conservative in the first place (see Subsection 3.5). In this sub-subsection, a Monte Carlo experiment is conducted to explore the effect of scaling $h = h^*_{\text{bind}}(\tau_n) = \frac{b-a}{\sqrt{\tau_n}}$, where $\tau_n = \lfloor n^{1/5} \rceil$, on the empirical pointwise bias.

Figure 9 depicts the empirical pointwise bias as a function of bandwidth h for the distributions in Figures 8, 15. Bias is computed from the empirical expectation taken over MC repetitions. In each repetition, a new n-large *i.i.d.* sample is drawn from f and then $h \mapsto \tilde{f}_{d_n,\tau_n}(x)$ is computed for different x. For computational efficiency, data are not encrypted. This is without loss of generality, as we saw previously that encryption does not really matter for the results, unlike the choice of τ_n, d_n. The bandwidth $h = h^*_{\text{bind}}(\tau_n) = \frac{b-a}{\sqrt{\tau_n}},\ \tau_n = \lfloor n^{1/5} \rceil$, is plotted with vertical dashed line. Thus, to left of the dashed line a less conservative, while to the right of it a more conservative bandwidth is used.

We can observe that the empirical bias of $\tilde{f}_{d_n,\tau_n}(x)$ when evaluated at the mean of the distribution is always smaller or the same for less conservative, smaller bandwidths presented here. This observation, however, does not carry over to other evaluation points: the empirical bias increases for smaller h for e.g. the median of the Generalised Pareto distribution, or the mean plus/minus two standard deviations for the von Mises distribution. Importantly, when h is 70% of the benchmark choice $\frac{b-a}{\sqrt{\tau_n}}$, the empirical bias at the 10% (or 90%) quantile of the Arcsine distribution increases sharply. It is because a too small h causes the inputs of the HE-kernel $\tilde{K}_{d,\tau}$ to fall outside the "sensible weighting" region (see Subsection 3.3). As a consequence, dissimilar observations to x start to get larger weights than similar ones. This leads to larger bias, especially for multimodal distributions when $\tilde{f}_{d_n,\tau_n}(x)$ is evaluated at an x near the boundary of the support of f.

Thus, we can conclude that while the bandwidth choice $h = h^*_{\text{bind}}(\tau_n) = \frac{b-a}{\sqrt{\tau_n}}$ might be too conservative for unimodal distributions, it is not conservative for multimodal distributions for evaluation points near the boundary of the support. Therefore, $h = h^*_{\text{bind}}(\tau_n) = \frac{b-a}{\sqrt{\tau_n}}$ is a safe choice ensuring "sensible weighting".

Figure 9: Empirical Bias of HE-KDE as a Function of Bandwidth



Notes: $\tau_n = \lfloor n^{1/5} \rceil$, $d_n = 14$, $n = 2^{15}$. $h = \frac{b-a}{\sqrt{\tau_n}}$ is indicated by the dashed, vertical black lines. Other h values of evaluation are 70, 80, 90, 95, 110, 130, 150, 170 percent of $h = \frac{b-a}{\sqrt{\tau_n}}$. q_α is the α-th quantile given by $F(q_\alpha) = \alpha$ for the cumulative distribution function F of the probability density function f. σ is the standard deviation of the VM distribution. Each line depicts the empirical bias, with empirical expectation computed from MC simulation with 20 repetitions on non-encrypted data, as a function of h, for a given x.

5. Conclusion

I proposed an estimator for univariate kernel density estimation (KDE), for bounded support probability density functions, on ring-homomorphically encrypted (HE) data, called HE-KDE. HE-KDE makes it possible for a client to outsource density estimation to a cloud without exposing the non-encrypted data to the cloud. To the best of my knowledge, this thesis is the first work to address (nonparametric) density estimation on encrypted data.

The encryption-related shortcomings of kernels typically used for KDE were discussed. HE-KDE fixes these issues by using a polynomial which locally approximates a kernel. HE-KDE routine is comprised of two parts: (i) the HE-adjusted estimator on non-encrypted data; and (ii) algorithms enabling its use with an HE-scheme on encrypted data.

In contrast to many HE papers, I carefully investigated the theoretical properties of the HE-adjusted estimator on non-encrypted data. The estimator is non-negative, and its integral is bounded by one. However, it is defective as the integral need not be one. Numerical analysis suggests that if, for increasing sample size (n), the degree of the approximate polynomial increases, the mean integrated square error (MISE) goes to zero faster than when the degree

does not increase. Increasing degree, however, implies weaker encryption security and longer computation time. Thus, there is a trade-off between the "goodness" of the estimator and both encryption security and computational costs.

A polynomial balancing these aspects was designed: an MISE-bound of order $O\left(n^{-0.5}\right)$ can be reached for sufficiently smooth, twice continuously differentiable densities, with support of length $\leq 10^6$, together with a security level $\lambda \leq \frac{7.2n}{1360} - 110$. This translates into a "reasonable" performance and a 63-bit security for $n = 2^{15} = 32768$. When the support is smaller or the density is smoother, the magnitude of the MISE-bound decreases and the convergence speed increases. Pseudo code for the algorithms enabling the use of the estimator with the CKKS (HEAAN) encryption scheme (Cheon et al., 2017) was provided, and usage examples presented. Numerical experiments suggest that "reasonable" performance might carry over to densities that are not continuous – hence not differentiable – everywhere on the real line.

References

Alia Alabdulkarim, Mznah Al-Rodhaan, Tinghuai Ma, and Yuan Tian. PPSDT: A Novel Privacy-Preserving Single Decision Tree Algorithm for Clinical Decision-Support Systems Using IoT Devices. *Sensors (Basel, Switzerland)*, 19, 2019.

Martin Albrecht, Melissa Chase, Hao Chen, Jintai Ding, Shafi Goldwasser, Sergey Gorbunov, Shai Halevi, Jeffrey Hoffstein, Kim Laine, Kristin Lauter, Satya Lokam, Daniele Micciancio, Dustin Moody, Travis Morrison, Amit Sahai, and Vinod Vaikuntanathan. Homomorphic Encryption Security Standard. Technical report, HomomorphicEncryption.org, Toronto, Canada, November 2018.

An Open Industry / Government / Academic Consortium to Advance Secure Computation. HomomorphicEncryption.org. URL `https://homomorphicencryption.org/`.

Song Bian, Masayuki Hiromoto, and Takashi Sato. Towards Practical Homomorphic Email Filtering: A Hardware-Accelerated Secure Naïve Bayesian Filter. In *Proceedings of the 24th Asia and South Pacific Design Automation Conference*, ASPDAC '19, page 621–626, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450360074. doi: 10.1145/3287624.3287699. URL `https://doi.org/10.1145/3287624.3287699`.

Raphael Bost, Raluca Popa, Stephen Tu, and Shafi Goldwasser. Machine Learning Classification over Encrypted Data. page 4324, 01 2015. doi: 10.14722/ndss.2015.23241.

Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. Fully Homomorphic Encryption without Bootstrapping. Cryptology ePrint Archive, Report 2011/277, 2011. URL `https://eprint.iacr.org/2011/277`.

Jung Cheon, Han Kyoohyung, Andrey Kim, Miran Kim, and Yongsoo Song. *Bootstrapping for Approximate Homomorphic Encryption*, pages 360–384. 01 2018. ISBN 978-3-319-78380-2. doi: 10.1007/978-3-319-78381-9_14.

Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 409–437. Springer, 2017.

Pedro M. Esperança, Louis J. M. Aslett, and Chris C. Holmes. Encrypted Accelerated Least Squares Regression. *ArXiv e-prints*, 2017.

Junfeng Fan and Frederik Vercauteren. Somewhat Practical Fully Homomorphic Encryption. Cryptology ePrint Archive, Report 2012/144, 2012. URL `https://eprint.iacr.org/2012/144`.

Craig Gentry. *A Fully Homomorphic Encryption Scheme*. PhD thesis, Stanford University, 2009. URL `https://crypto.stanford.edu/craig`.

Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. Cryptology ePrint Archive, Report 2013/340, 2013. URL https://eprint.iacr.org/2013/340.

Francisco-Javier González-Serrano, Adrián Amor-Martín, and Jorge Casamayón-Antón. Supervised Machine Learning Using Encrypted Training Data. *International Journal of Information Security*, 17(4):365–377, 2018.

Thore Graepel, Kristin Lauter, and Michael Naehrig. ML Confidential: Machine Learning on Encrypted Data. In *International Conference on Information Security and Cryptology*, pages 1–21. Springer, 2012.

Xiaoqian Jiang, Miran Kim, Kristin Lauter, and Yongsoo Song. Secure Outsourced Matrix Computation and Application to Neural Networks. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 1209–1222, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356930. doi: 10.1145/3243734.3243837. URL https://doi.org/10.1145/3243734.3243837.

Charles F. F. Karney. Sampling Exactly from the Normal Distribution. *ACM Transactions on Mathematical Software*, 42(1):1–14, Mar 2016. ISSN 1557-7295. doi: 10.1145/2710016. URL http://dx.doi.org/10.1145/2710016.

Alhassan Khedr, Glenn Gulak, and Vinod Vaikuntanathan. SHIELD: Scalable Homomorphic Implementation of Encrypted Data-Classifiers. *IEEE Transactions on Computers*, 65(9):2848–2858, 2015. naiver bayes classifier using gsw2013.

Andrey Kim, Yongsoo Song, Miran Kim, Keewoo Lee, and Jung Hee Cheon. Logistic Regression Model Training Based on the Approximate Homomorphic Encryption. *BMC Medical Genomics*, 11(4):83, 2018a.

Miran Kim, Yongsoo Song, Shuang Wang, Yuhou Xia, and Xiaoqian Jiang. Secure Logistic Regression Based on Homomorphic Encryption: Design and Evaluation. *JMIR Medical Informatics*, 6(2):e19, Apr 2018b. ISSN 2291-9694. doi: 10.2196/medinform.8805. URL http://medinform.jmir.org/2018/2/e19/.

Alexandre Leblanc. On Estimating Distribution Functions Using Bernstein Polynomials. *Annals of the Institute of Statistical Mathematics*, 64(5):919–943, October 2012. doi: 10.1007/s10463-011-0339-4. URL https://ideas.repec.org/a/spr/aistmt/v64y2012i5p919-943.html.

Adriana Lopez-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-Fly Multiparty Computation on the Cloud via Multikey Fully Homomorphic Encryption. Cryptology ePrint Archive, Report 2013/094, 2013. URL https://eprint.iacr.org/2013/094.

Wenjie Lu, Yoshiji Yamada, and Jun Sakuma. Efficient Secure Outsourcing of Genome-wide Association Studies. In *2015 IEEE Security and Privacy Workshops*, pages 3–6. IEEE, 2015.

Isa Muqattash and Mohammed Yahdi. Infinite Family of Approximations of the Digamma Function. *Mathematical and Computer Modelling*, 43(11):1329 – 1336, 2006. ISSN 0895-7177. doi: https://doi.org/10.1016/j.mcm.2005.02.010. URL http://www.sciencedirect.com/science/article/pii/S0895717705004735.

Adrian Pagan and Aman Ullah. *Nonparametric Econometrics*. Themes in Modern Econometrics. Cambridge University Press, 1999. doi: 10.1017/CBO9780511612503.

Pascal Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, pages 223–238, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg. ISBN 978-3-540-48910-8.

Saerom Park, Junyoung Byun, Joohee Lee, Jung-Hee Cheon, and Jaewook Lee. HE-friendly Algorithm for Privacy-Preserving SVM Training. *IEEE Access*, PP:1–1, 03 2020. doi: 10.1109/ACCESS.2020.2981818.

Anna Poon, Steve Jankly, and Tingting Chen. Privacy Preserving Fisher's Exact Test on Genomic Data. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 2546–2553. IEEE, 2018.

Yogachandran Rahulamathavn, Raphael Phan, Suresh Veluru, Kanapathippillai Cumanan, and Muttukrishnan Rajarajan. Privacy-Preserving Multi-Class Support Vector Machine for Outsourcing the Data Classification in Cloud. *Dependable and Secure Computing, IEEE Transactions on*, 11:467–479, 09 2014. doi: 10.1109/TDSC.2013.51.

Baek Kyung Song, Joon Soo Yoo, Miyeon Hong, and Ji Won Yoon. A Bitwise Design and Implementation for Privacy-Preserving Data Mining: From Atomic Operations to Advanced Algorithms. *Security and Communication Networks*, 2019, 2019. doi: 10.1155/2019/3648671.

A.W. van der Vaart. Mathematische Statistiek Lecture Notes, 2002.

Y. Yang, X. Huang, X. Liu, H. Cheng, J. Weng, X. Luo, and V. Chang. A Comprehensive Survey on Secure Outsourced Computation and Its Applications. *IEEE Access*, 7:159426–159465, 10 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2949782.

Joon Soo Yoo, Jeong Hwan Hwang, Baek Kyung Song, and Ji Won Yoon. A Bitwise Logistic Regression Using Binary Approximation and Real Number Division in Homomorphic Encryption Scheme. In Swee-Huay Heng and Javier Lopez, editors, *Information Security Practice and Experience*, pages 20–40, Cham, 2019. Springer International Publishing. ISBN 978-3-030-34339-2.

Yuchen Zhang, Wenrui Dai, Xiaoqian Jiang, Hongkai Xiong, and Shuang Wang. FORESEE: Fully Outsourced secuRe gEnome Study basEd on Homomorphic Encryption. In *BMC medical informatics and decision making*, volume 15, page S5. Springer, 2015.

Appendices

A. Proofs

A.1. Proposition 1

Proof. Table 1 records the cumulative number of multiplications ciphertexts $\boldsymbol{c}_{\text{power}}$ and $\boldsymbol{c}_{\text{term}}$) are involved in when the given line of Algorithm 2 is finished executing.

The recording starts at line 10. By line 8, $\boldsymbol{c}_{\text{power}}$ is involved in 2 multiplications via $\boldsymbol{c}$: multiplication with $\check{h}$ and squaring. Thus $\boldsymbol{c}_{\text{power}}$ starts from 2 multiplications. At line 9, $\boldsymbol{c}_{\text{term}}$ is derived from $\boldsymbol{c}_{\text{power}}$ by a multiplication, thus it start from $2+1=3$ multiplications.

In each iteration of the for-loop, $\boldsymbol{c}_{\text{power}}$ is involved in one additional multiplication. $\boldsymbol{c}_{\text{term}}$ is derived from $\boldsymbol{c}_{\text{power}}$ by another multiplication so it has one more multiplication than $\boldsymbol{c}_{\text{power}}$ by line 13. There are $d-1$ repetitions in the for-loop. Hence when the last repetition is completed, $\boldsymbol{c}_{\text{power}}$ is involved in $2+(d-1)$ multiplications, thus $\boldsymbol{c}_{\text{term}}$ is involved in $2+(d-1)+1=d+2$.

The ciphertexts $\boldsymbol{c}$, $\boldsymbol{c}_{\text{result}}$ do not participate in further multiplications. Hence we conclude that the multiplicative depth of Algorithm 2 is $d+2$ as was to be shown.

Table 1: Cumulative Number of HE Multiplications in Algorithm 2

Line	$\boldsymbol{c}_{\text{power}}$	$\boldsymbol{c}_{\text{term}}$
Before loop, line 10	2	3
Loop $k=2$, line 12	3 (+1)	3
Loop $k=2$, line 13	3 $\Longrightarrow$	4 (+1)
Loop $k=3$, line 12	4 (+1)	4
Loop $k=3$, line 13	4 $\Longrightarrow$	5 (+1)
$\vdots$	$\vdots$	$\vdots$
Loop $k=d$, line 12	$2+(d-1)$ (+1)	$2+(d-1)$
Loop $k=d$, line 13	$2+(d-1)$ $\Longrightarrow$	$2+(d-1)+1$ (+1)

Notes: (+1) indicates the change in the number of multiplications.

■

A.2. Proposition 2

Proof. (i) $\tilde{f}_{d,\tau}(x) \geq 0$. It is sufficient to show that $\tilde{K}_{d,\tau}(u) \geq 0 \ \forall u \in [\pm\sqrt{\tau}]$, which I do by showing non-negativity of the unnormalised kernel $u \mapsto \gamma + \sum_{k=1}^{d} w_k(1+u^2-\tau)^k$. Rewrite this as $t \mapsto \gamma + \sum_{k=1}^{d} w_k t^k$ where $t \coloneqq 1+u^2-\tau \in [1-\tau, 1]$. I partition the domain $[1-\tau,1] = [1-\tau,0] \cup [0,1]$ and consider the partitions separately.

When $t \in [1-\tau, 0]$, then for all odd k, $w_k < 0$ by definition, hence $w_k t^k \geq 0$. For all even k, $w_k > 0$ (see Figure 10), hence $w_k t^k \geq 0$. Thus $t \mapsto \gamma + \sum_{k=1}^{d} w_k t^k \geq 0 \ \forall t \in [1-\tau, 0]$ because $\gamma \approx 0.577 > 0$.

When $t \in [0, 1]$, I visually inspect $t \mapsto \gamma + \sum_{k=1}^{d} w_k t^k$ for different even degrees. Figure 10 depicts the function $t \mapsto \min_{d \in \mathcal{D}} \gamma + \sum_{k=1}^{d} w_k t^k$, for $\mathcal{D} \coloneqq \{2, 4, \ldots, 10000\}$ and $t \in [0, 1]$. This function is always positive, thus $t \mapsto \gamma + \sum_{k=1}^{d} w_k t^k \geq 0 \ \forall t \in [0, 1]$. To conclude, as the unnormalised kernel is always positive, the normalising constant $c_{d,\tau}$ is always positive, hence $\tilde{K}_{d,\tau}(u) \geq 0 \ \forall u \in [\pm\sqrt{\tau}], \ \forall \tau \in \mathbb{R}_{\geq 1}$, and for all even $d \in \mathbb{N}$.

(ii) $\int_S \tilde{f}_{d,\tau}(x)dx \leq 1$. By definition, $\int_S \tilde{f}_{d\tau}(x)dx = \int_a^b \frac{1}{nh} \sum_{i \in [n]} \tilde{K}_{d,\tau}\left(\frac{x - x_i}{h}\right) dx$. Using Tonelli's theorem to interchange summation and integral,

$$\int_a^b \frac{1}{nh} \sum_{i \in [n]} \tilde{K}_{d,\tau}\left(\frac{x - x_i}{h}\right) dx = \sum_{i \in [n]} \frac{1}{nh} \int_a^b \tilde{K}_{d,\tau}\left(\frac{x - x_i}{h}\right) dx.$$

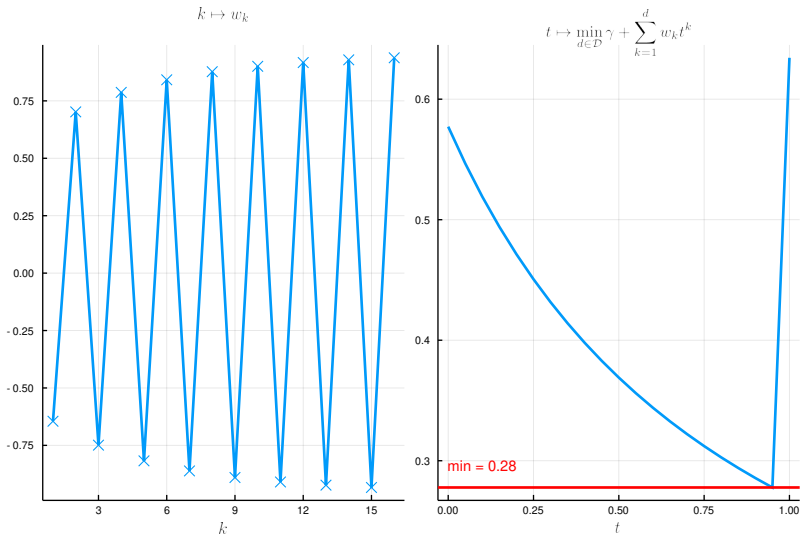
After a change of variable $z \coloneqq \frac{x - x_i}{h}$, we can upper bound the integral by increasing its upper, and decreasing its lower limit, since $\tilde{K}_{d,\tau}(z) \geq 0$ for all $z \in [\pm\sqrt{\tau}] \supset [\frac{a-b}{h}, \frac{b-a}{h}] \supset [\frac{a-x_i}{h}, \frac{b-x_i}{h}] \ \forall i \in [n]$ because $x, x_i \in [a, b]$ and $h \geq \frac{b-a}{\sqrt{\tau}}$.

$$\sum_{i \in [n]} \frac{1}{nh} \int_{\frac{a-x_i}{h}}^{\frac{b-x_i}{h}} \tilde{K}_{d,\tau}(z)\, h dz \leq \sum_{i \in [n]} \frac{1}{n} \int_{\frac{a-b}{h}}^{\frac{b-a}{h}} \tilde{K}_{d,\tau}(z)\, dz \leq \sum_{i \in [n]} \frac{1}{n} \int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z) dz = 1$$

where the last step $(= 1)$ follows from construction of $\tilde{K}_{d,\tau}$.

■

Figure 10: Local Approximate Kernel: Sign Properties



Notes: in the right figure, a step size of 0.05 is used for t for evaluation.

A.3. Proposition 3

Proof. (i) $\mathbb{V}\left[\tilde{f}_{d,\tau}(x)\right]$. By definition, $\mathbb{V}\left[\tilde{f}_{d,\tau}(x)\right] = \mathbb{V}\left[\frac{1}{nh}\sum_{i\in[n]}\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)\right]$. By Tonelli's theorem and that $(x_i)_{i\in[n]}$ is *i.i.d.*, we have

$$\mathbb{V}\left[\frac{1}{nh}\sum_{i\in[n]}\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)\right] = \frac{1}{n^2h^2}\sum_{i\in[n]}\mathbb{V}\left[\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)\right] = \frac{1}{nh^2}\mathbb{V}\left[\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)\right]$$

for some $i \in [n]$. Note that $\mathbb{V}\left[\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)\right] = \mathbb{E}\left[\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)^2\right] - \mathbb{E}\left[\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)\right]^2 \leq \mathbb{E}\left[\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)^2\right]$, where

$$\begin{aligned}\mathbb{E}\left[\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)^2\right] &= \int_a^b \tilde{K}_{d,\tau}\left(\frac{x-t}{h}\right)^2 f(t)dt \\ &= \int_{\frac{x-a}{h}}^{\frac{x-b}{h}} \tilde{K}_{d,\tau}(z)^2 f(x-hz)(-h)dz = h\int_{\frac{x-b}{h}}^{\frac{x-a}{h}} \tilde{K}_{d,\tau}(z)^2 f(x-hz)dz\end{aligned}$$

with a change of variables $z \coloneqq \frac{x-t}{h}$. We can change the limit of the integral by noticing that $f(x-hz) = 0$ whenever $x - hz \notin [a,b]$ (because the support of f is $[a,b]$), which happens if either $z > \frac{x-a}{h}$ or $z < \frac{x-b}{h}$. Provided that $\tilde{K}_{d,\tau}(z)^2 < \infty$ when $z \in [\pm\sqrt{\tau}]$, $\tilde{K}_{d,\tau}(z)^2 f(x-hz) = 0$ when $z \in (\frac{x-a}{h}, \sqrt{\tau})$ or $z(-\sqrt{\tau}, \frac{x-b}{h})$. Because $x \in [a,b]$ and $h \leq \frac{b-a}{\sqrt{\tau}}$, both of these intervals are well-defined: $\frac{x-a}{h} \leq \sqrt{\tau}$ and $-\sqrt{\tau} \leq \frac{x-b}{h}$. Thus, $\int_{\frac{x-b}{h}}^{\frac{x-a}{h}} \tilde{K}_{d,\tau}(z)^2 f(x-hz)dz = \int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)^2 f(x-hz)dz$. Last, apply Taylor-expansion to $f(x-hz)$ around x to obtain

$$\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)^2 f(x-hz)dz = \int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)^2 \left[f(x) - hzf'(x) + o(h)\right]dz =: (nh)\cdot V(x,n,h,d,\tau)$$

Thus,

$$\mathbb{V}\left[\tilde{f}_{d,\tau}(x)\right] \leq \frac{1}{nh^2}\mathbb{E}\left[\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)^2\right] \leq V(x,n,h,d,\tau) \approx \frac{f(x)}{nh} I_1(d,\tau).$$

for n large, h small.

(ii) $\mathbb{B}\mathrm{ias}(\tilde{f}_{d,\tau}(x))$. By definition $\mathbb{B}\mathrm{ias}(\tilde{f}_{d,\tau}(x)) = \mathbb{E}\left[\tilde{f}_{d,\tau}(x)\right] - f(x)$. Using Tonelli's theorem and that $(x_i)_{i\in[n]}$ is *i.i.d.*, we have

$$\mathbb{E}\left[\tilde{f}_{d,\tau}(x)\right] = \mathbb{E}\left[\frac{1}{nh}\sum_{i\in[n]}\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)\right] = \frac{1}{h}\mathbb{E}\left[\tilde{K}_{d,\tau}\left(\frac{x-x_i}{h}\right)\right] = \frac{1}{h}\int_a^b \tilde{K}_{d,\tau}\left(\frac{x-t}{h}\right)f(t)dt.$$

The change of variables $z \coloneqq \frac{x-t}{h}$ yields $\frac{1}{h}\int_a^b \tilde{K}_{d,\tau}\left(\frac{x-t}{h}\right)f(t)dt = \frac{-h}{h}\int_{\frac{x-a}{h}}^{\frac{x-b}{h}} \tilde{K}_{d,\tau}(z)\,f(x-hz)dz = \int_{\frac{x-b}{h}}^{\frac{x-a}{h}} \tilde{K}_{d,\tau}(z)\,f(x-hz)dz$.

Again, we can change the limit of the integral by noticing that (ii.1) $f(x-hz) = 0$ whenever $x - hz \notin [a,b]$ (because the support of f is $[a,b]$), which happens if either

$z > \frac{x-a}{h}$ or $z < \frac{x-b}{h}$; and that (ii.2) $\tilde{K}_{d,\tau}(z) < \infty$ when $z \in [\pm\sqrt{\tau}]$ because $\tilde{K}_{d,\tau}(z) \geq 0$ when $z \in [\pm\sqrt{\tau}]$ and $\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)\,dz = 1$. As a consequence, $\tilde{K}_{d,\tau}(z)\,f(x-hz) = 0$ when $z \in (\frac{x-a}{h}, \sqrt{\tau})$ or $z(-\sqrt{\tau}, \frac{x-b}{h})$. Thus, changing limit and applying Taylor expansion to $f(x-hz)$ around x gives

$$\begin{aligned}\mathbb{E}\left[\tilde{f}_{d,\tau}(x)\right] &= \int_{\frac{x-b}{h}}^{\frac{x-a}{h}} \tilde{K}_{d,\tau}(z)\,f(x-hz)dz = \int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)\,f(x-hz)dz \\ &= \int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)\left[f(x) - f'(x)hz + \frac{f''(x)}{2}(hz)^2 + o(h^2)\right]dz.\end{aligned}$$

Using that $\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)\,dz = 1$ and that $\int_{-\sqrt{\tau}}^{\sqrt{\tau}} z^k \tilde{K}_{d,\tau}(z)\,dz = 0$ for all odd $k \in \mathbb{N}$ due to symmetry of $\tilde{K}_{d,\tau}$, yields

$$\mathbb{B}\text{ias}(\tilde{f}_{d,\tau}(x)) = \frac{h^2 f''(x)}{2} I_2(d,\tau) + o(h^2).$$

(iii) MISE$(\tilde{f}_{d,\tau}(x))$. I follow the proof of van der Vaart (2002). By definition MISE$(\tilde{f}_{d,\tau}(x)) = \int_a^b \mathbb{E}\left[(\tilde{f}_{d,\tau}(x) - f(x))^2\right]dx = \int_a^b \left(\mathbb{V}\left[\tilde{f}_{d,\tau}(x)\right] + \mathbb{B}\text{ias}(\tilde{f}_{d,\tau}(x))^2\right)dx$. Below I compute the integrated variance and squared bias.

Integrated variance. Using results from (i) and Tonelli's theorem,

$$\int_a^b \mathbb{V}\left[\tilde{f}_{d,\tau}(x)\right]dx \leq \int_a^b \frac{1}{nh}\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)^2\,f(x-hz)dzdx = \frac{1}{nh}\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)^2 \int_a^b f(x-hz)dxdz.$$

Change variables $t := x - hz$, so that $\int_a^b f(x-hz)dx = \int_{a-hz}^{b-hz} f(t)dt \leq 1$ because $b - hz > a - hz$ and f has support $[a,b]$. Therefore, $\int_a^b \mathbb{V}\left[\tilde{f}_{d,\tau}(x)\right]dx \leq \frac{1}{nh}\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)^2 = \frac{I_1(d,\tau)}{nh}$.

Integrated squared bias. Based on (ii), $\mathbb{E}\left[\tilde{f}_{d,\tau}(x)\right] = \int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)\,f(x-hz)dz$, so that we can write $\mathbb{B}\text{ias}(\tilde{f}_{d,\tau}(x)) = \mathbb{E}\left[\tilde{f}_{d,\tau}(x)\right] - f(x) = \int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)\,(f(x-hz) - f(x))dz$ because $\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)\,dz = 1$. Applying Taylor expansion to $f(x-hz)$ with the Laplacian representation of the remainder yields

$$\begin{aligned}\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)\,(f(x-hz) - f(x))dz &= \int_{-\sqrt{\tau}}^{\sqrt{\tau}}\int_0^1 \tilde{K}_{d,\tau}(z)\left[-hzf'(x) - (hz)^2 f''(x-shz)(1-s)\right]dsdz \\ &= -h^2\int_{-\sqrt{\tau}}^{\sqrt{\tau}}\int_0^1 \tilde{K}_{d,\tau}(z)\underbrace{z}_{:=U}\underbrace{zf''(x-shz)(1-s)}_{:=V}dsdz\end{aligned}$$

as $\int_{-\sqrt{\tau}}^{\sqrt{\tau}} z\tilde{K}_{d,\tau}(z)\,dz = 0$ due to symmetry of $\tilde{K}_{d,\tau}$. The double integral can be thought of as an expectation $\mathbb{E}[UV]$ of independent random variables U with density $\tilde{K}_{d,\tau}$ and s uniformly distributed on [0,1] with density 1. By Cauchy-Schwartz inequality $\mathbb{E}[UV]^2 \leq$

$\mathbb{E}[U^2]\,\mathbb{E}[V^2]$. Therefore we can upper bound the term above as

$$\begin{aligned}\mathbb{B}\mathrm{ias}(\tilde{f}_{d,\tau}(x))^2 &\leq h^4 \underbrace{\left(\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \tilde{K}_{d,\tau}(z)\, z^2 dz\right)}_{\mathbb{E}[U^2]} \underbrace{\left(\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \int_0^1 \tilde{K}_{d,\tau}(z)\, z^2 f''(x-shx)^2(1-s)^2 ds dz\right)}_{\mathbb{E}[V^2]} \\ &= h^4 I_2(d,\tau) \left(\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \int_0^1 \tilde{K}_{d,\tau}(z)\, z^2 f''(x-shx)^2(1-s)^2 ds dz\right).\end{aligned}$$

Hence

$$\begin{aligned}\int_a^b \mathbb{B}\mathrm{ias}(\tilde{f}_{d,\tau}(x))^2 dx &\leq h^4 I_2(d,\tau) \left(\int_{-\sqrt{\tau}}^{\sqrt{\tau}} \int_0^1 \tilde{K}_{d,\tau}(z)\, z^2 \int_a^b f''(x-shx)^2 dx (1-s)^2 ds dz\right) \\ &\leq h^4 I_2(d,\tau)^2 \frac{\beta_f}{3}\end{aligned}$$

where the last step follows from the assumption $\int_S f''(x-\beta)dx \leq \beta_f$ and that $\int_0^1 (1-s)^2 ds = 1/3$ is the second moment of the uniformly distributed random variable $1-s$ on $[0,1]$. Therefore,

$$\begin{aligned}\mathrm{MISE}(\tilde{f}_{d,\tau}) &= \int_a^b \left(\mathbb{V}\left[\tilde{f}_{d,\tau}(x)\right] + \mathbb{B}\mathrm{ias}(\tilde{f}_{d,\tau}(x))^2\right) dx \\ &\leq \frac{I_1(d,\tau)}{nh} + h^4 I_2(d,\tau)^2 \frac{\beta_f}{3}.\end{aligned}$$

■

A.4. Proposition 4

Proof. $M_{d,\tau} : \mathbb{R}_{++} \to \mathbb{R}_+$ is twice differentiable and its second derivative is $M''_{d,\tau}(h) = \frac{2h_1(d,\tau)}{nh^3} + 12h^2 h_2(d,\tau)$. For every non-trivial $\tilde{K}_{d,\tau}$, the functions $h_1(d,\tau), h_2(d,\tau) > 0$, hence $M''_{d,\tau}(h) > 0$ for all $h \in \mathbb{R}_{++}$. That is, $M_{d,\tau}$ is strictly convex, obtaining a global minimum on $\mathbb{R}_{++}$ at the critical point $h^*_{\text{nonbind}} : M'_{d,\tau}(h^*_{\text{nonbind}}) = 0$. It follows that $h^*_{\text{nonbind}} = \left(\frac{h_1(d,\tau)}{4h_2(d,\tau)n}\right)^{1/5} =: h^*_{\text{nonbind}}(d,\tau)$, and the global minimum is $M_{d,\tau}(h^*_{\text{nonbind}}(d,\tau)) = (2^7 h_2(d,\tau) h_1(d,\tau)^4 n^{-4})^{1/5}$.

Consider imposing the linear constraint $h \geq \frac{b-a}{\sqrt{\tau}}$. If $h^*_{\text{nonbind}}(d,\tau) > \frac{b-a}{\sqrt{\tau}} =: h^*_{\text{bind}}(\tau)$, then the constraint is not binding because $h^*_{\text{nonbind}}(d,\tau)$ is the global minimiser. If $h^*_{\text{nonbind}}(d,\tau) \leq h^*_{\text{bind}}(\tau)$, the constraint is binding because $M_{d,\tau}(h)$ is strictly convex and $\lim_{h\to\infty} M_{d,\tau}(h) = \infty$. Therefore, the constraint is binding if and only if $h^*_{\text{nonbind}}(d,\tau) \leq h^*_{\text{bind}}(\tau)$. In that case the objective value is $M_{d,\tau}(h^*_{\text{bind}}(\tau)) = \frac{\sqrt{\tau} h_1(d,\tau)}{n(b-a)} + \frac{(b-a)^4}{\tau^2} h_2(d,\tau)$. Because $M_{d,\tau}(h^*_{\text{nonbind}}(d,\tau)) = \min_{h\in\mathbb{R}_{++}} M_{d,\tau}(h)$ and $M_{d,\tau}(h^*_{\text{bind}}(\tau)) = \min_{h\in\mathbb{R}_{++}:h\geq\frac{b-a}{\sqrt{\tau}}} M_{d,\tau}(h)$, we have $M_{d,\tau}(h^*_{\text{bind}}(\tau))$ $\geq M_{d,\tau}(h^*_{\text{nonbind}}(d,\tau))$ as $\{h \in \mathbb{R}_{++} : h \geq \frac{b-a}{\sqrt{\tau}}\} \subset \mathbb{R}_{++}$. ■

B. Algorithms

Algorithm 4 Sum

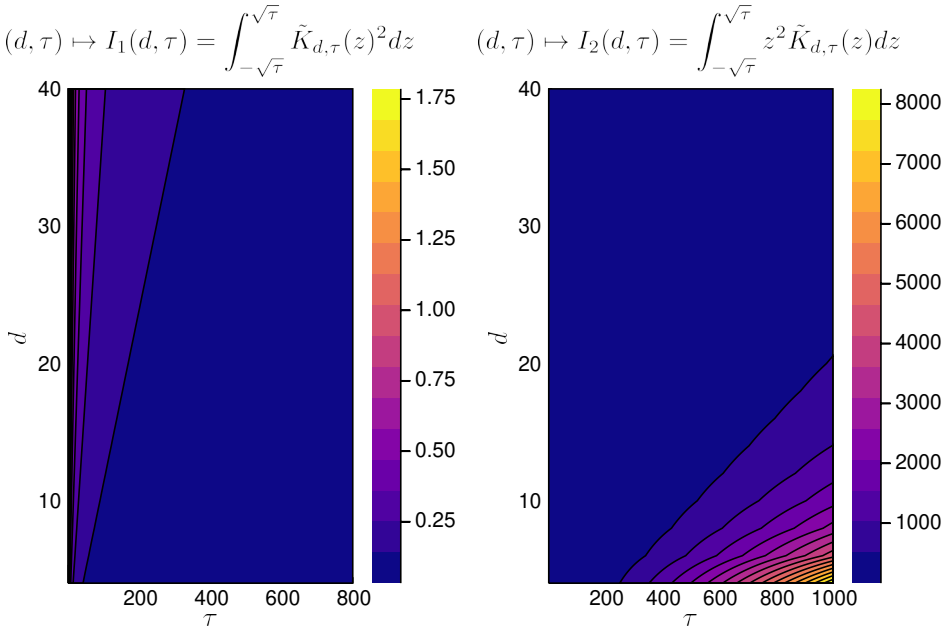
input: $\boldsymbol{c}$ encrypting (the encoding of) $(z_j)_{j\in T}$ under secret key sk; switching keys $(\mathsf{swk}_{i,j})_{i,j\in T}$ where $\mathsf{swk}_{i,j} = \mathsf{KSGen}(\kappa_k(\mathsf{sk}))$ with $k = j^{-1}i \pmod{2N}$

output: $\boldsymbol{c}_s$ encrypting (the encoding of) $(\hat{z}_j)_{j\in T}$, the first element of which is $\sum_{j\in T} z_j$

let $\Psi : T \to \{1, 2, \ldots, N/2\}$ be an invertible mapping (e.g. enumerate T)
$\boldsymbol{c}_s \coloneqq \boldsymbol{c}$
for $i \in \{2, \ldots, N/2\}$ **do**
    $\boldsymbol{c}_{\text{perm}} \coloneqq \mathsf{Perm}^{\Psi^{-1}(i)\to\Psi^{-1}(1)}_{\mathsf{swk}_{\Psi^{-1}(i),\Psi^{-1}(1)}}(\boldsymbol{c})$ ▷ map i-th element to first element
    $\boldsymbol{c}_s \coloneqq \mathsf{Add}(\boldsymbol{c}_s, \boldsymbol{c}_{\text{perm}})$ ▷ encrypting a vector with first element $\sum_{j=1}^{i} z_{\Psi^{-1}(j)}$
return $\boldsymbol{c}_s$

C. Figures

Figure 11: Integrals I_1, I_2 of Proposition 3



C.1. Parameter Choices

In the following Sub-subsections C.1.1 and C.1.2, I illustrate the impact of S and β_f on the convergence of MISE-bound, continuing the discussion on statistical and encryption optimality in Subsection 3.5. I set $a := 0$ and fix β_f. Then I replicate Figures 5 and 6 for different values of b. Without loss of generality, this corresponds to increasing S or β_f.

C.1.1 Statistical Optimality

For $h = \frac{b-a}{\sqrt{\tau_n}}, \tau_n = \lfloor n^{2/5} \rceil, d_n = \lfloor n^{4/5} \rceil_2$, Figure 12 replicates Figure 5 for different values of b. Two conclusions can be drawn.

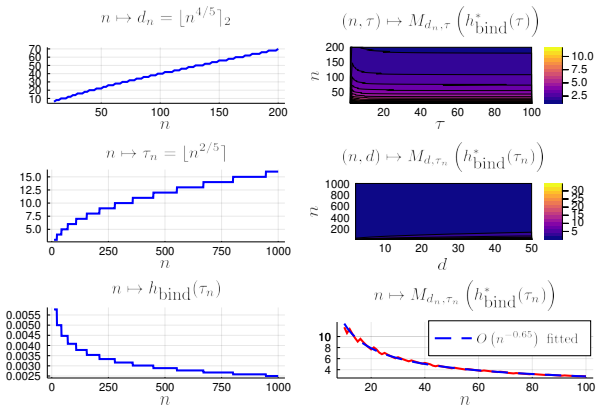
First, the larger is the support S (or β_f), the larger is the MISE-bound. This follows from the fact that $M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n))\big|_{h^*_{\text{bind}}(\tau_n)=\frac{b-a}{\sqrt{\tau_n}}} = \frac{h_1(d_n,\tau_n)}{n^{4/5}(b-a)} + \frac{(b-a)^4}{n^{4/5}} h_2(d_n,\tau_n)$ for $\tau_n = n^{2/5}$. Larger S leads to higher bandwidth h, which increases the term corresponding to the squared bias, $\frac{(b-a)^4}{n^{4/5}} h_2(d_n,\tau_n)$. Intuitively, larger support forces distant observations closer than they should be so that they fall in the "sensible weighting" region of the kernel. This leads to larger bias. Larger β_f means that the density f "changes a lot" in its support, therefore over-weighting (h not small enough) leads to larger bias.

Second, the larger is the support S (or β_f), the faster is the convergence of MISE-bound to zero. Larger S increases the influence of the second (bias) term $\frac{(b-a)^4}{n^{4/5}} h_2(d_n,\tau_n)$ on the convergence compared to the first one $\frac{h_1(d_n,\tau_n)}{n^{4/5}(b-a)}$. The second term is the one d_n mostly affects, via

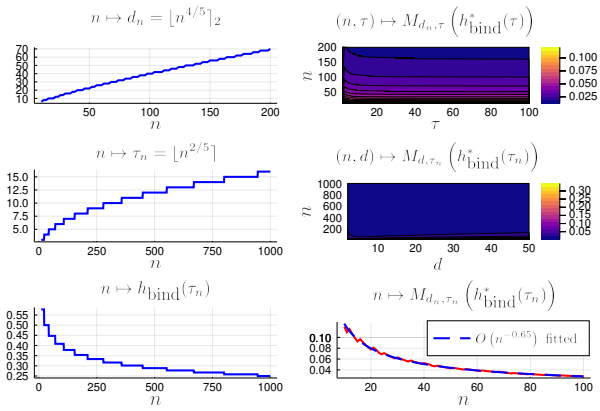
I_2 (see Subsection 3.5), and this effect is strong as suggested by Figure 11. As a consequence, we witness a faster convergence. This mechanism is visible in the first and second row, second column plots in each of Figure 12. As S increases, the smaller role τ plays in lowering $M_{d_n,\tau}(h^*_{\text{bind}}(\tau))$; and the larger role d plays in lowering $M_{d,\tau_n}(h^*_{\text{bind}}(\tau_n))$.

Figure 12: Statistically Optimal Parameter Choices for Different S

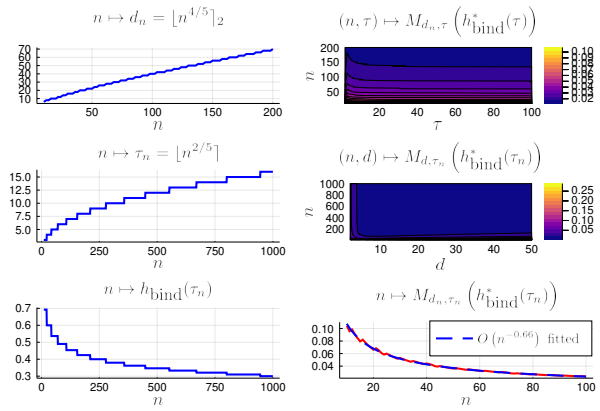
(a) $S = [0, 0.01]$, $\beta_f = 3$



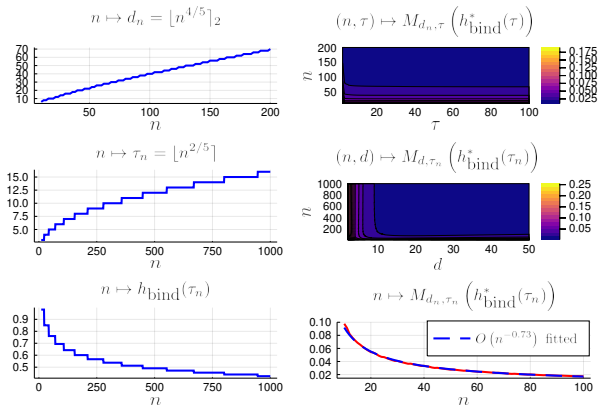
(b) $S = [0, 1]$, $\beta_f = 3$



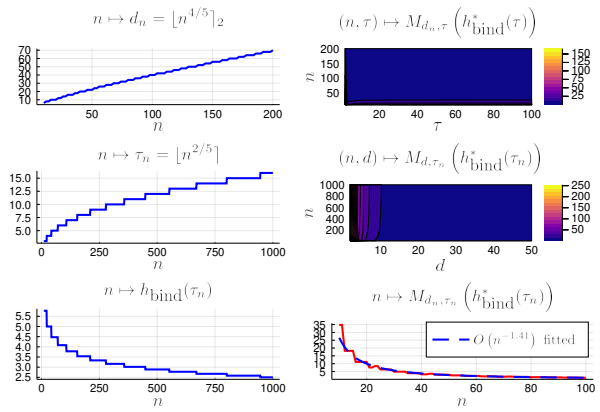
(c) $S = [0, 1.2]$, $\beta_f = 3$



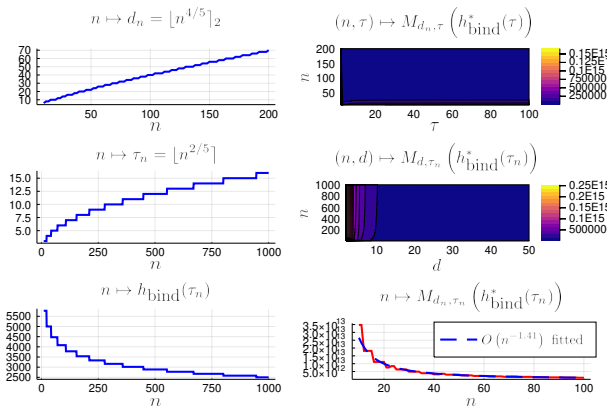
(d) $S = [0, 1.7]$, $\beta_f = 3$



(e) $S = [0, 10]$, $\beta_f = 3$



(f) $S = [0, 10000]$, $\beta_f = 3$



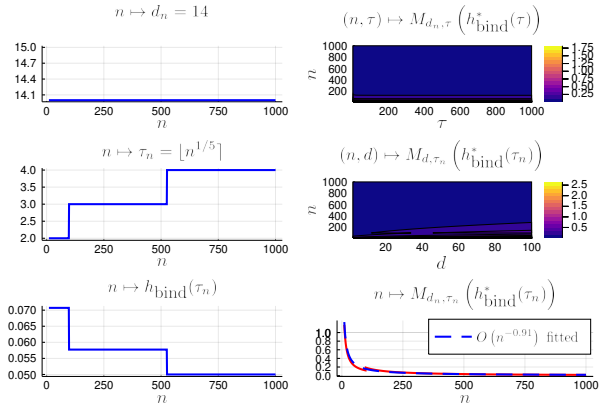
Notes: fitted line obtained by estimating the model $y_n = \alpha_0 n^{\alpha_1} \varepsilon_n$ with ordinary least squares after log - log transform for $y_n := M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n))$, some error term ε_n, and model parameters α_0, α_1.

C.1.2 Encryption Optimality

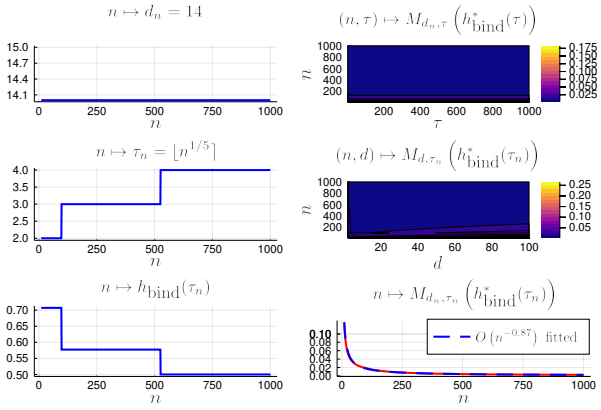
For $h = \frac{b-a}{\sqrt{\tau_n}}, \tau_n = \lfloor n^{1/5} \rceil, d_n = 14$, Figure 12 replicates Figure 13 for different values of b. The first conclusion in C.1.1 continues to hold. That is, for larger S (or β_f), the MISE-bound is also larger. This is because $M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n))\big|_{h^*_{\text{bind}}(\tau_n)} = \frac{h_1(d_n,\tau_n)}{n^{9/10}(b-a)} + \frac{(b-a)^4 h_2(d_n,\tau_n)}{\tau_n^{2/5}}$ for $\tau_n = \lfloor n^{1/5} \rceil, d_n = 14$. The second conclusion in C.1.2 does not hold anymore. That is, for larger S (of β_f), the speed of convergence becomes slower rather than faster. This is expected – it is due to the same reason as faster convergence in C.1.1. Namely, larger S leads to increased importance of the second, bias term in $M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n))$. While an increasing d_n in C.1.1 decreased the bias term, the constant $d_n = 14$ does not have such an effect. However, there is convergence, even for $S = [0, 10^6]$, and, importantly, it is achieved together with an acceptable level of encryption security λ.

Figure 13: Encryption-optimal Parameter Choices for Different S

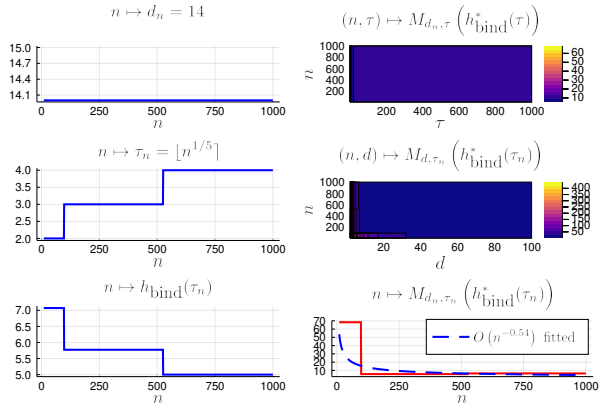
(a) $S = [0, 0.1]$, $\beta_f = 3$



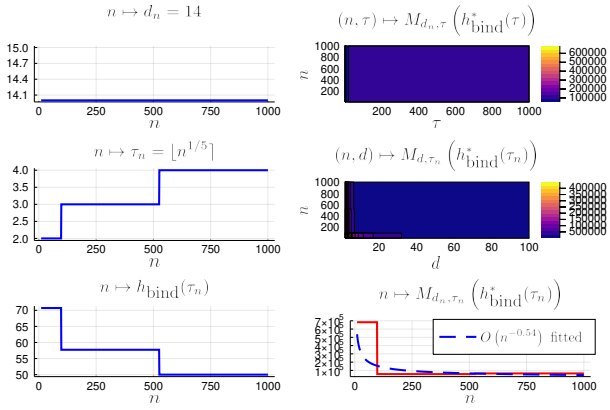
(b) $S = [0, 1]$, $\beta_f = 3$



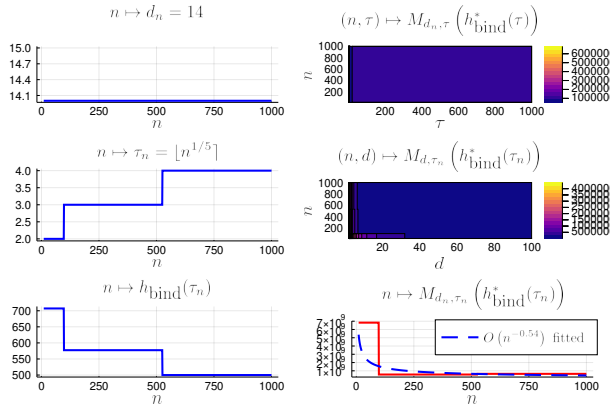
(c) $S = [0, 10]$, $\beta_f = 3$



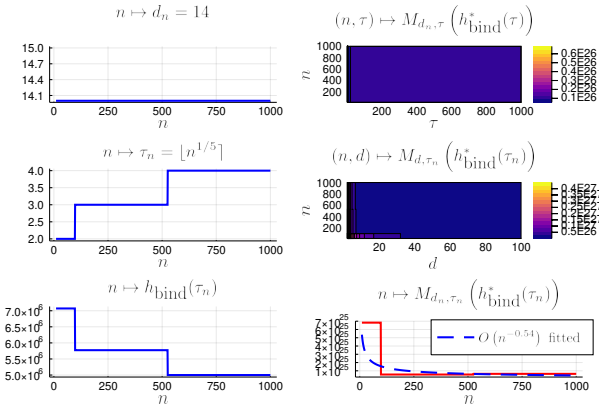
(d) $S = [0, 100]$, $\beta_f = 3$



(e) $S = [0, 1000]$, $\beta_f = 3$



(f) $S = [0, 1000000]$, $\beta_f = 3$



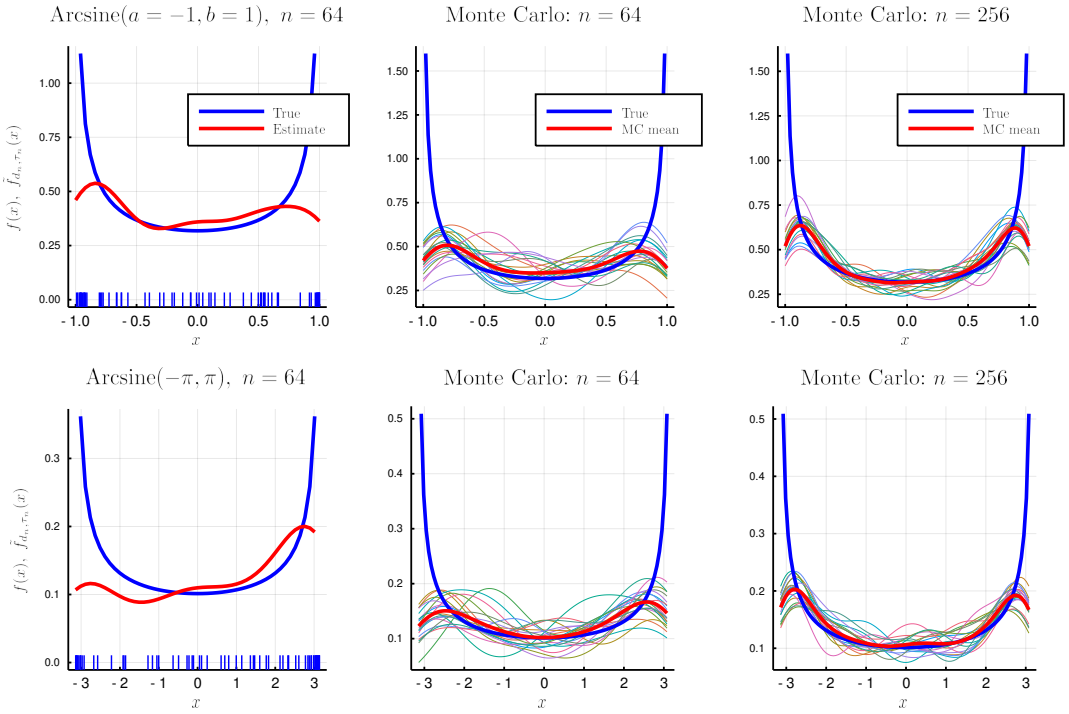
Notes: fitted line obtained by estimating the model $y_n = \alpha_0 n^{\alpha_1} \varepsilon_n$ with ordinary least squares after log - log transform for $y_n \coloneqq M_{d_n,\tau_n}(h^*_{\text{bind}}(\tau_n))$, some error term ε_n, and model parameters α_0, α_1.

C.2. HE-KDE Examples

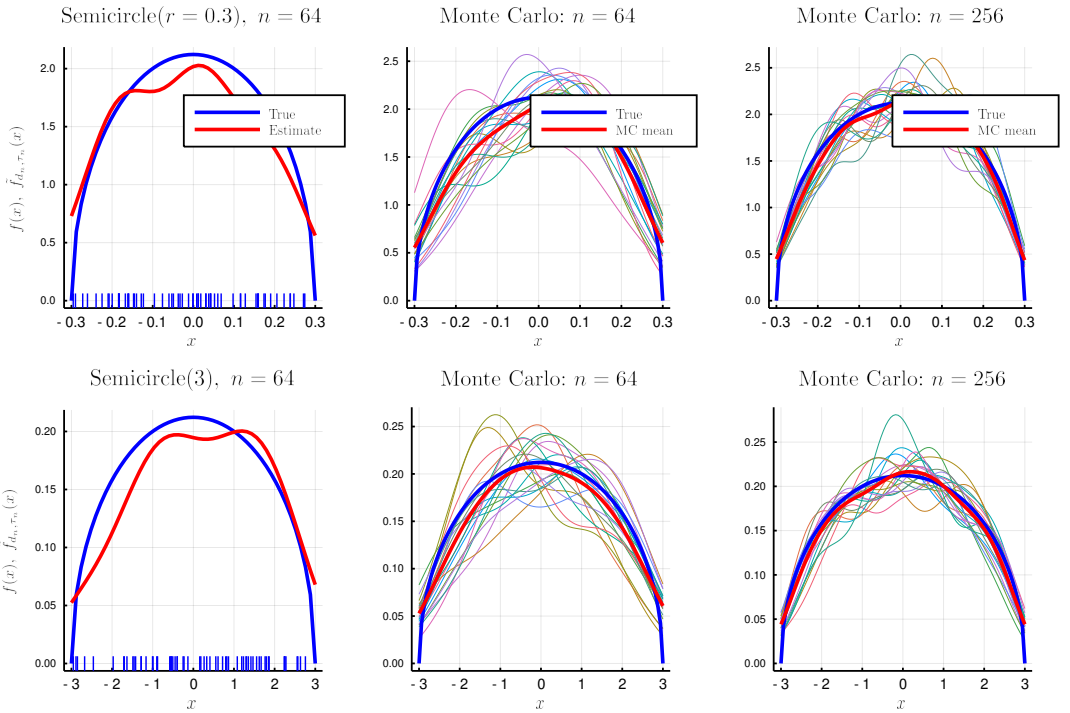
C.2.1 Non-encrypted Data

Figure 14: HE-KDE More Non-encrypted Examples

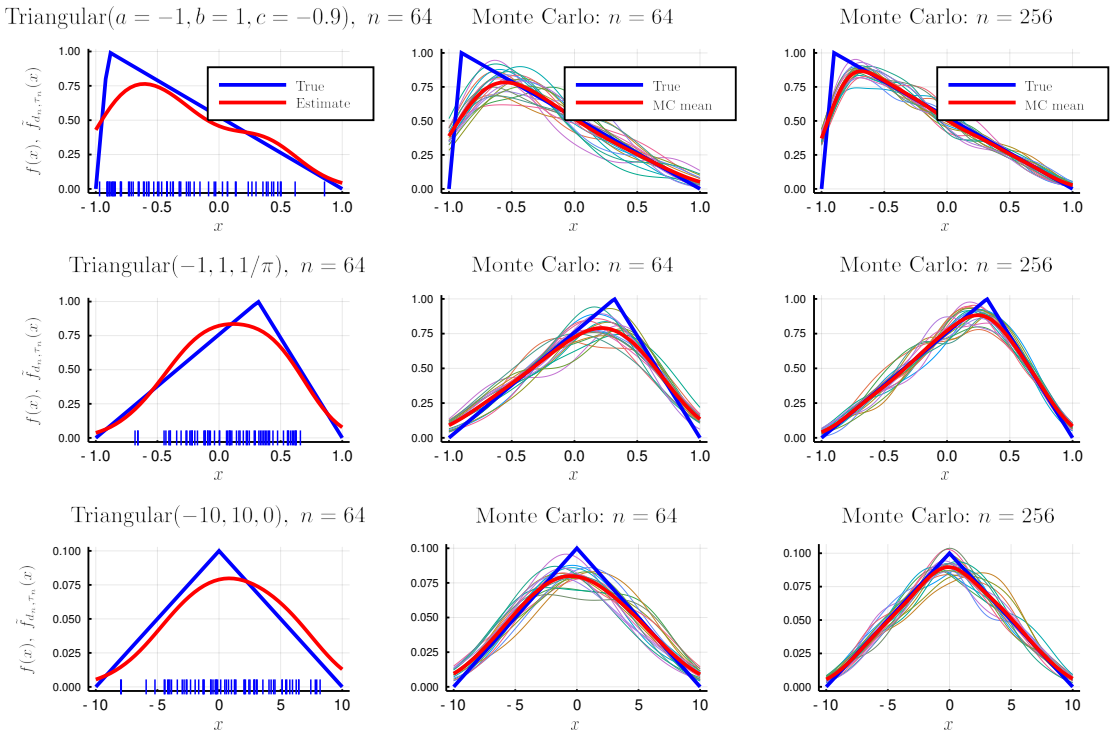
(a) Arcsine Distribution



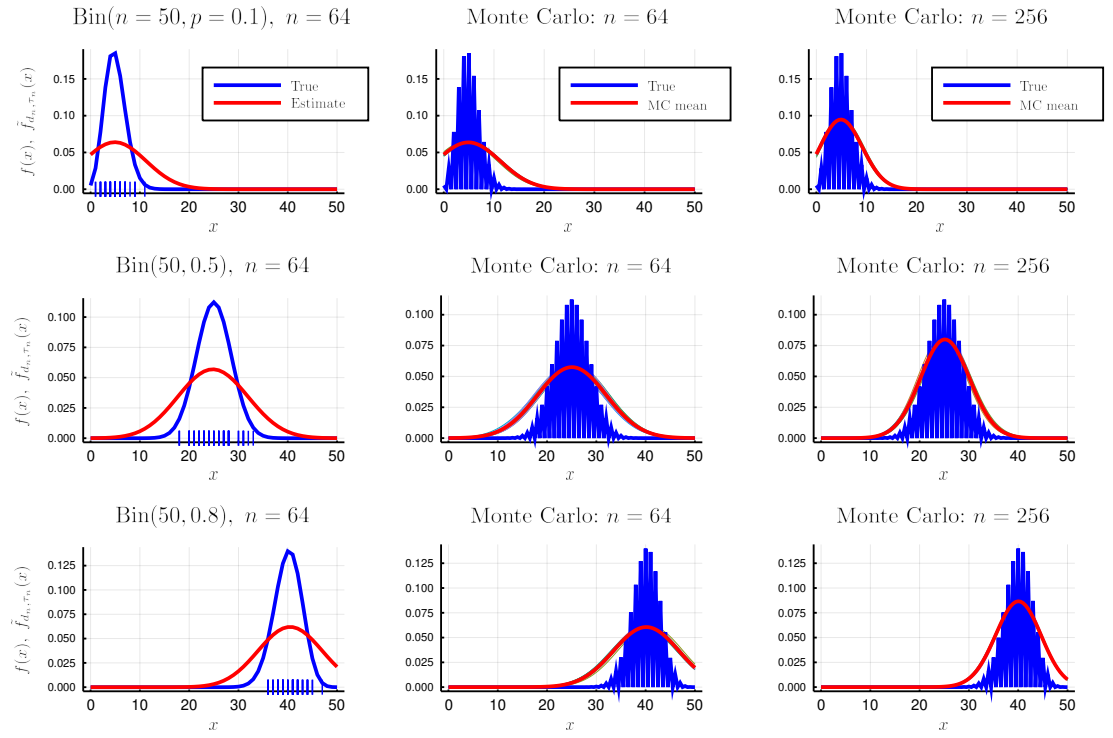
(b) Semicircle Distribution



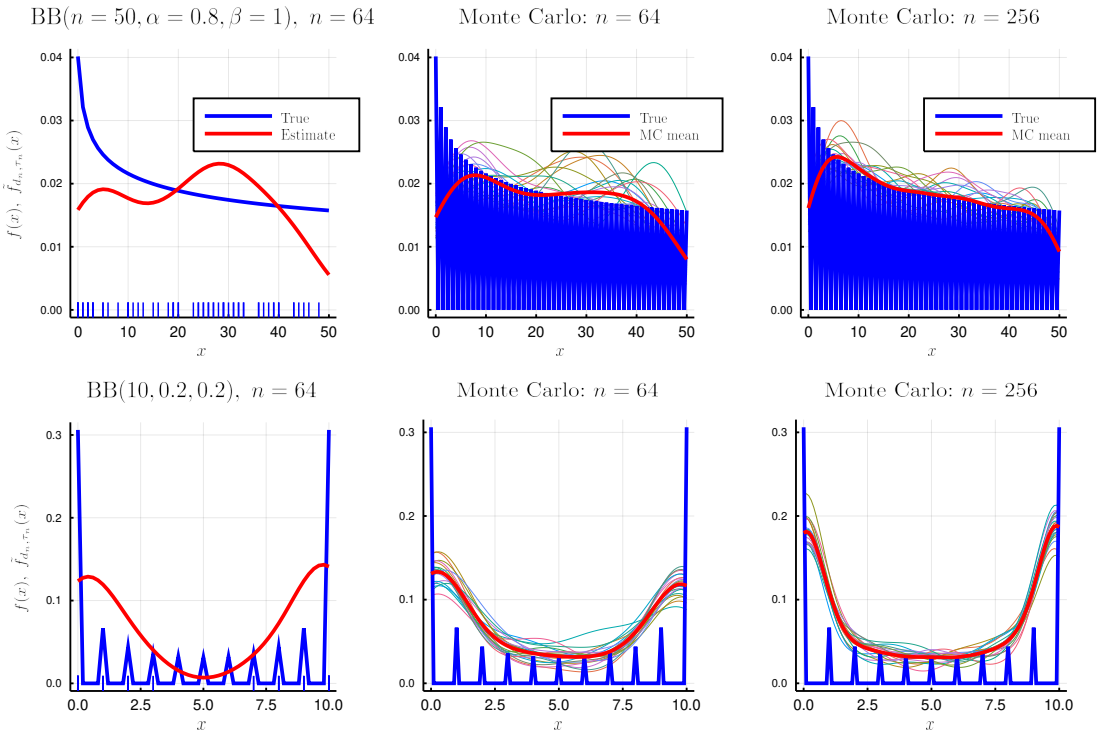
(c) Triangular Distribution



(d) Binomial Discrete Distribution



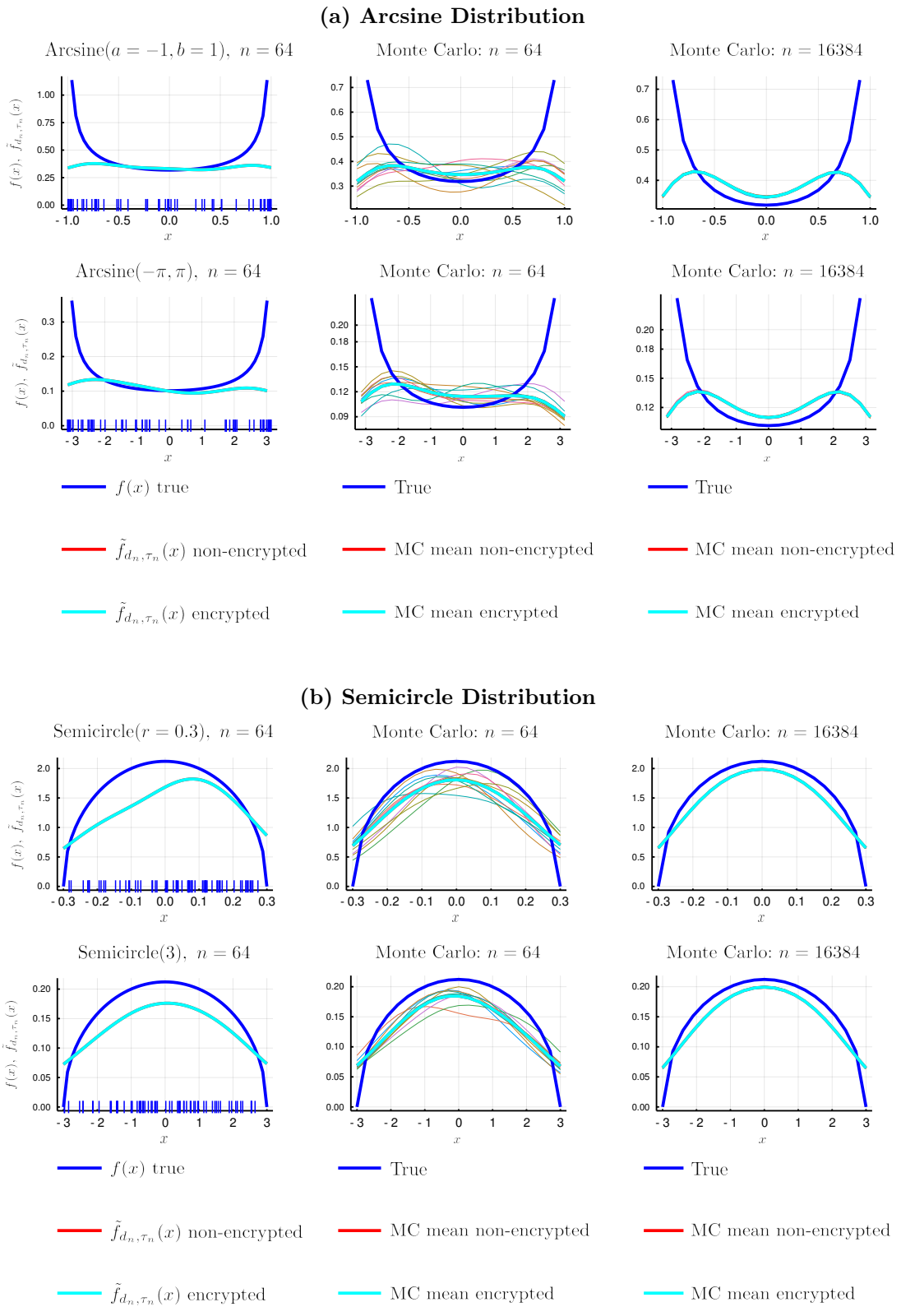
(e) Beta-binomial Discrete Distribution



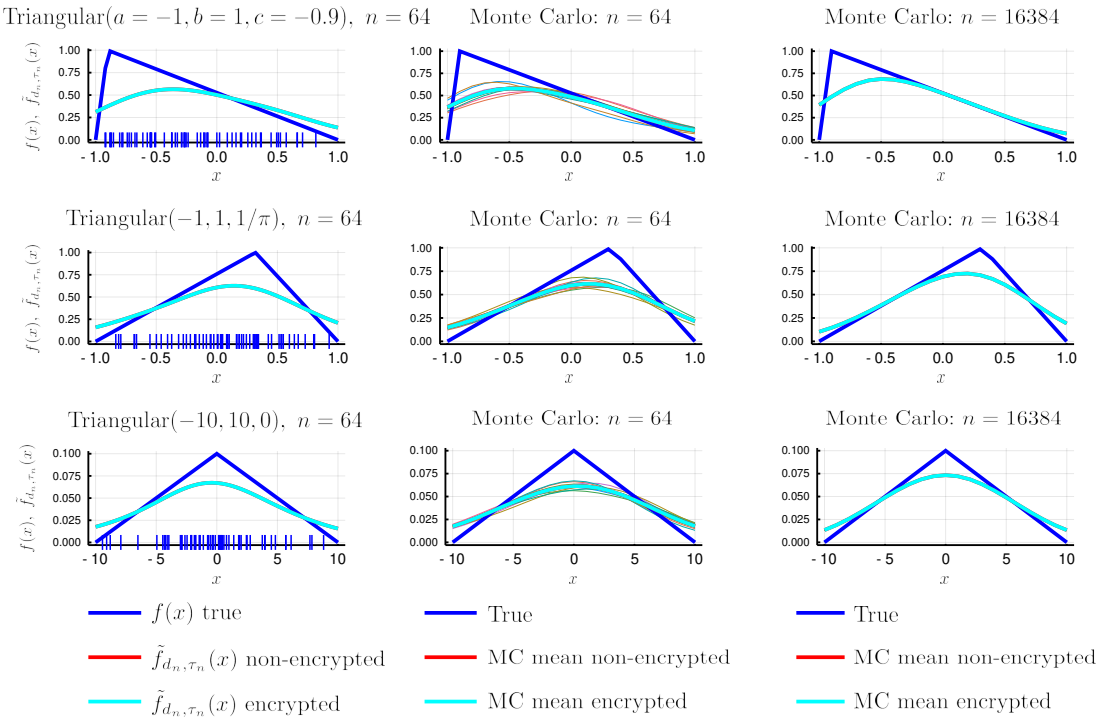
Notes: $h = \frac{b-a}{\sqrt{\tau_n}}$, $\tau_n = \lfloor n^{2/5} \rceil$, $d_n = \lfloor n^{4/5} \rceil_2$. First column: Non-encrypted HE-KDE estimate $\tilde{f}_{d_n,\tau_n}$ for a single n-large sample from true density f. Sample values indicated on the x-axis with "|". Second and third columns: Monte Carlo (MC) simulations. Each thin line, depicting non-encrypted $x \mapsto \tilde{f}_{d_n,\tau_n}(x)$, corresponds to a MC repetition. There are 20 repetitions. Thick red line is the mean of $x \mapsto \tilde{f}_{d_n,\tau_n}(x)$ across MC repetitions. Each row corresponds to different f-parameters (in brackets in the first column). f and $\tilde{f}_{d_n,\tau_n}$ are evaluated at 100 equidistant points.

C.2.2 Encrypted Data

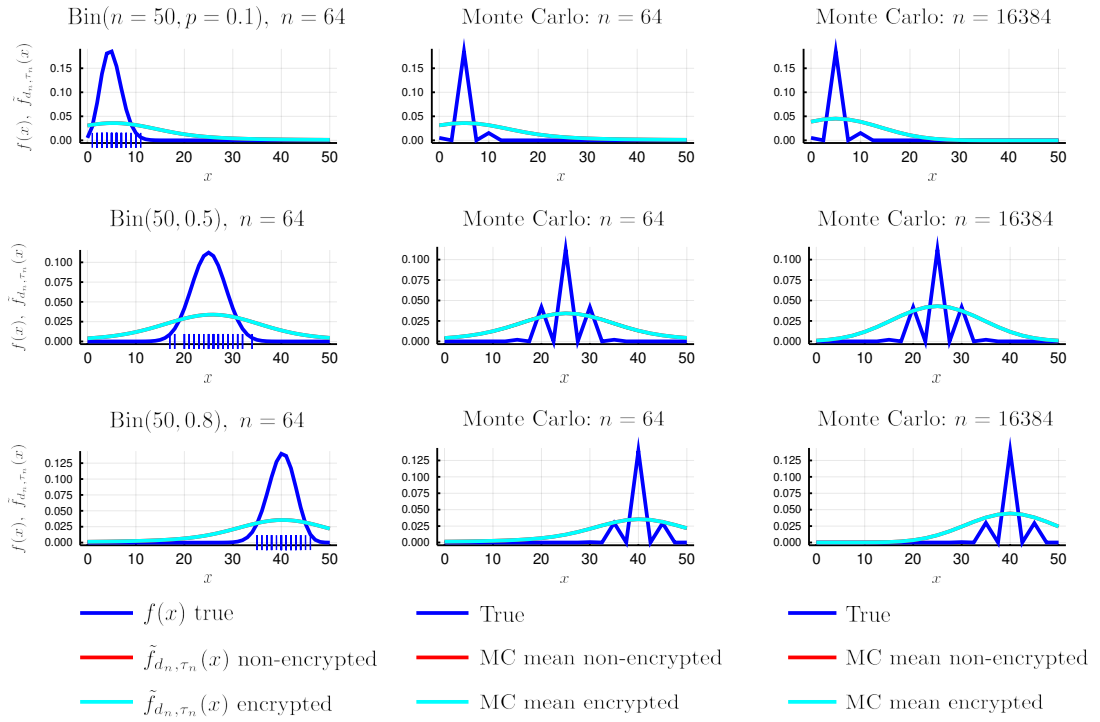
Figure 15: HE-KDE More Encrypted Examples



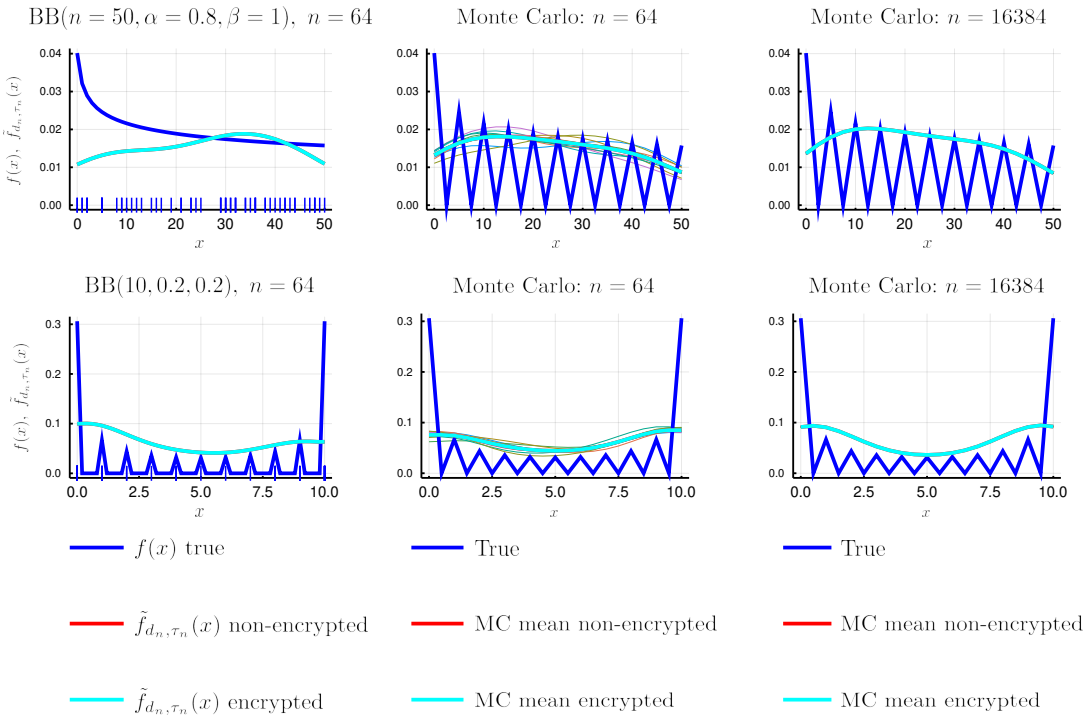
(c) Triangular Distribution



(d) Binomial Discrete Distribution



(e) Beta-binomial Discrete Distribution



Notes: $h = \frac{b-a}{\sqrt{\tau_n}}$, $\tau_n = \lfloor n^{1/5} \rceil$, $d_n = 14$. First column: HE-KDE estimate $\tilde{f}_{d_n,\tau_n}$ (non-encrypted and encrypted) for a single n-large sample from true density f. Non-encrypted sample values indicated on the x-axis with "|". Second and third columns: Monte Carlo (MC) simulations. Each thin line, depicting $\tilde{f}_{d_n,\tau_n}$, corresponds to a MC repetition on encrypted data. Thick cyan line is the mean of $\tilde{f}_{d_n,\tau_n}(x)$ across MC repetitions on encrypted data. Thick red line is the mean of $\tilde{f}_{d_n,\tau_n}(x)$ across MC repetitions on non-encrypted data (not plotted). There are 10 repetitions. Each row corresponds to different f-parameters (in brackets in the first column). In the first column, f and $\tilde{f}_{d_n,\tau_n}$ are evaluated at 50, in the other columns, at 20 equidistant points.