

# Approximate Decryption in Homomorphic Division and Privacy Impact

**Ameerah Alsulami**

Florida Institute of Technology  
Melbourne, FL, USA  
King Khalid University  
Abha, Saudi Arabia  
aalsulami2019@my.fit.edu  
aalsulami@kku.edu.sa

**Marius Silaghi**

Florida Institute of Technology  
Melbourne, FL, USA  
msilaghi@fit.edu

## Abstract

The result of a computation over secret inputs inherently reveals some information about those inputs; such **semantic leakage** is unavoidable. The challenge is to ensure that the computation method does not introduce additional, avoidable disclosure beyond what is implied by the output itself. This issue is particularly critical in privacy-preserving machine learning and cloud-based data processing, where homomorphic encryption enables computation over encrypted data but often relies on practical approximations.

Division-enabled homomorphic encryption schemes based on rational encodings preserve arithmetic correctness, but their decrypted outputs may retain sufficient algebraic structure to enable inference of the original operands, creating representation-induced leakage.

We study **approximate decryption** as a privacy-aware interpretation mechanism for homomorphic division, by combining **symmetric additive masking** with **continued fraction expansion** to recover meaningful approximations while avoiding exact reconstruction. We empirically compare **Shared- $k$**  and **Distinct** ( $k_a, k_b$ ) with respect to numerical growth and reconstruction accuracy under **approximate decryption**, showing that the latter achieves smoother growth and lower reconstruction failure. This work identifies a previously underexplored privacy risk and demonstrates that approximation-based decryption provides a practical mitigation in settings where bounded numerical error is acceptable.

## Introduction

Privacy-preserving computation is a fundamental requirement in cloud-based analytics and machine learning, where sensitive data must be processed without direct exposure. Homomorphic encryption (HE) enables computation directly over encrypted data and can reduce disclosure risk in outsourced computation environments (Gentry 2009; Acar et al. 2018; Yi et al. 2014). Despite its strong theoretical guarantees, fully homomorphic encryption (FHE) remains computationally expensive for many real-world deployments (Brakerski, Gentry, and Vaikuntanathan 2014; Acar et al. 2018; Chen et al. 2017). Consequently, partially

homomorphic encryption (PHE) schemes are frequently adopted, as they efficiently support restricted arithmetic operations while providing practical privacy guarantees (Fan and Vercauteren 2012).

A prominent line of work extends ElGamal-based multiplicative homomorphism to support *division* by encoding plaintext values as fixed-point rational numbers and encrypting the numerator and denominator independently (ElGamal 1985). Although arithmetic correctness is preserved, such constructions may remain fully *invertible*: namely, decrypting the division output can directly reveal the original operands through preserved numerical structure, thereby undermining confidentiality (Sidorov, Wei, and Ng 2022).

To mitigate this limitation, we study an approximation-based privacy enhancement introducing *secret additive masking* terms to prevent exact reconstruction by factoring at decryption time. Further, improved fidelity recovery of application-acceptable approximation are shown possible via continued fraction expansion. Practical applications—including encrypted sensing and privacy-preserving machine learning—naturally tolerate bounded numerical error (Cheon et al. 2017), allowing approximation to suppress deterministic leakage while preserving usability. This perspective emphasizes practical reconstruction behavior and observable leakage patterns arising from representation structure.

After introducing background, subsequent sections detail the proposed approximation schemes and corresponding recovery mechanisms, before example cases are analyzed and conclusions are drawn.

## Background

ElGamal encryption provides multiplicative homomorphism (ElGamal 1985). Given public key  $(p, g, y)$ , where  $p$  is a large prime,  $g$  a number of large order  $\pmod{p}$ , and  $y$  computed as  $g^s \pmod{p}$  for the secret  $s$ , encryption of some message  $m$  is

$$E(m) = (m \cdot y^x \pmod{p}, g^x \pmod{p}), \quad (1)$$

performed with a fresh random secret  $x$  per encryption, where  $\gcd(x, p-1) = 1$ . To enable division (Sidorov, Wei, and Ng 2022), real values  $a$  are encoded as

$$E'(a) = (E(a_1), E(a_2)) \quad (2)$$

where  $a_1$  and  $a_2$  in Equation 2 are defined using some sufficiently large constant  $C$  as:

$$a_1 = a \cdot C \quad (3)$$

$$a_2 = C \quad (4)$$

In the original proposal, it was proposed to select  $C = 10^k$ , a value easy to understand but less optimal for masking purposes, yielding.

$$a_1 = a \cdot 10^k \quad (5)$$

$$a_2 = 10^k \quad (6)$$

If secret  $b$  is hidden as  $E(b) = (E(b_1), (b_2))$ , secret division of  $a$  by  $b$  is computed via cross-multiplication:

$$E'(a/b) = (E(a_1)E(b_2), E(a_2)E(b_1)) \quad (7)$$

$$= (E(a_1b_2), E(a_2b_1)). \quad (8)$$

At decryption, the numerator and denominator are revealed independently (Chung and Kim 2018), enabling exact recovery of  $a$  and  $b$  in many cases based of factoring, mainly when some partial information is already available.

**Homomorphic Division with Masking.** To mask factors, random factors  $c$  are thrown in the mix with the formula (Silaghi and Alsulami 2023):

$$\frac{E(a_1) E(b_2)}{E(a_2) E(b_1)} = \frac{E(a_1b_2)}{E(a_2b_1)} = \frac{E(a_1b_2c)}{E(a_2b_1c)} \quad (9)$$

The constructions above explains why division outputs can still carry deterministic structure. The next section discusses how masking breaks this structure and why approximation becomes a natural solution.

### Analysis of Challenges

In certain cryptographic applications, such as deterministic encryption or multiplicative blinding, structural patterns in rational values may inadvertently leak secrets, posing a potential privacy risk. To mitigate this, we introduce additive perturbations that disrupt post-decryption algebraic linkage based on factoring in rational division outputs. Unlike reversible randomness used for exact cancellation, our approach treats the additive term(s) as unknown at the time of decryption and interprets the decrypted ratio by reversing the approximation using continuous fractions methods.

Here, we examine an approximation-oriented instantiation and additionally present an alternative perturbation pattern that targets the same leakage source through different numerical behavior. Solutions aim to suppress deterministic post-decryption structure; they differ in how numerical growth and approximation behavior manifest, which motivates the later empirical study. This design shifts the focus from exact reconstruction toward controlled interpretation of decrypted outputs in application-relevant settings.

This strategy is particularly effective where exact reconstruction is unnecessary. Domains such as encrypted sensor data aggregation and privacy-preserving machine learning often tolerate bounded approximation error, making approximation-based interpretation practically meaningful under such constraints (Khinchin and Teichmann 1964; Boemer et al. 2019).

### Encryption Process with Symmetric Additive Masking

To hide factors of parameters, we inject symmetric additive noise  $c_1, c_2 \in \mathbb{Z}_p^*$  across the encoded components.

$$E'(a_1/a_2) \approx (E(a_1 \cdot 10^k + c_1), E(a_2 \cdot 10^k - c_2))$$

These perturbation should return a slightly larger number, thereby making up for the opposite rounding possible at recovery with partial fraction decomposition. Simpler alternatives modify either only the numerator or only the denominator:

$$E'(a_1/a_2) \approx (E(a_1 \cdot 10^k + c_1), E(a_2 \cdot 10^k))$$

$$E''(a_1/a_2) \approx (E(a_1 \cdot 10^k), E(a_2 \cdot 10^k - c_2))$$

where  $0 < c_1, c_2 \ll 10^k$ . Real values are encoded using fixed-point representation:

$$a_1 = a \cdot 10^k, \quad a_2 = 10^k$$

**Numerical Example.** Let  $a = 5$ ,  $b = 6$ , and choose  $k = 4$ , so  $10^k = 10000$ .

$$a_1 = 5 \cdot 10000 = 50000, \quad a_2 = 10000$$

$$b_1 = 6 \cdot 10000 = 60000, \quad b_2 = 10000$$

Encryption with additive masking using  $c_{a1} = 3$  and  $c_{b1} = 3$  yields:

$$E(a) = \frac{E(a_1 + c_{a1})}{E(a_2)} = \frac{E(50003)}{E(10000)}$$

$$E(b) = \frac{E(b_1 + c_{b1})}{E(b_2)} = \frac{E(60003)}{E(10000)}$$

Apply multiplicative masking using  $c = 2$ :

$$E(a/b) = \frac{E(50003)}{E(10000)} \cdot \frac{E(10000)}{E(60003)}$$

$$= \frac{E(50003) E(10000) E(2)}{E(10000) E(60003) E(2)} = \frac{E(1,000,060,000)}{E(1,200,060,000)}$$

Prior fraction-based encrypted division schemes preserve a rigid algebraic structure at the ciphertext level, which may encode sufficient relational information to allow east inference of the original operands after decryption; for example, scaled ciphertext ratios such as  $5,000,000/6,000,000$  directly reflect the plaintext values  $a = 5$  and  $b = 6$ .

In contrast, the symmetric additive masking perturbs both the numerator and denominator prior to division, producing the encrypted ratio  $E(1,000,060,000)/E(1,200,060,000)$ , which no longer preserves an exact scaled relationship between the encoded operands. By breaking this deterministic structure at the encryption stage, the scheme suppresses direct algebraic linkage between ciphertext components while remaining compatible with homomorphic division.

## Approximate Decryption via Continued Fraction Expansion

Recovery of the exact fraction devoid of revealing factors may still be possible using continued fraction decomposition, even if it is not always guaranteed (Khinchin and Teichmann 1964; Hardy and Wright 1979; Kelsey, Schneier, and Wagner 1996). Once exact recoverability is no longer the absolute objective, the remaining question is how masked division outputs can be best reconditioned and meaningfully interpreted. This lead us to approximation-based decryption via continued fraction reconstruction.

Continued fraction decomposition can help recover the exact intended ratio but that is not guaranteed. Fortunately, in many practical workloads, exact numerical recovery is not strictly required. Applications such as encrypted sensing, aggregation, and privacy-preserving machine learning inference naturally tolerate bounded numerical error. Within these settings, selecting a low-denominator convergent under a predefined threshold yields outputs that remain semantically meaningful, while avoiding disclosure of the exact underlying operands.

Given an encoded pair  $(m, n)$ , we aim to recover the rational number  $r = \frac{m}{n}$  by expressing it as a continued fraction, and recognizing the correct interpretation based on the problem characteristics.

### Step 1: Construct Rational Number

From the pair  $(m, n)$ , form the rational number:

$$r = \frac{m}{n}$$

### Step 2: Continued Fraction Expansion

We express  $r$  in the form of a continued fraction:

$$r = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \dots}}}$$

The terms  $a_0, a_1, a_2, \dots$  are determined as follows:

$$a_0 = \left\lfloor \frac{m}{n} \right\rfloor, r_1 = \frac{1}{\frac{m}{n} - a_0}, a_1 = \lfloor r_1 \rfloor, r_2 = \frac{1}{r_1 - a_1}, a_2 = \lfloor r_2 \rfloor, \dots$$

Each term adds to the continued structure using a '+' in the fractional layers:

$$r = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \dots}}}$$

### Step 3: Reconstructing the Rational from Continued Fraction

To verify or recreate the rational, use the convergent formula:

$$\frac{p_k}{q_k} = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{\dots + \frac{1}{a_k}}}}$$

The numerators  $p_k$  and denominators  $q_k$  are computed recursively:

$$\begin{aligned} p_{-1} &= 1, & p_0 &= a_0, & p_k &= a_k p_{k-1} + p_{k-2} \\ q_{-1} &= 0, & q_0 &= 1, & q_k &= a_k q_{k-1} + q_{k-2} \end{aligned}$$

### Step 4: Final Output

Knowledge of the problem domain allows for the selected convergent  $\frac{p_k}{q_k}$  to yield a good approximation of the original rational value. If the result is exact, then:

$$\frac{p_k}{q_k} = \frac{m}{n}$$

### Sample Decryption

$$D(m, n) \rightarrow 1,000,060,000, 1,200,060,000$$

First, reduce the rational by its greatest common divisor:

$$\frac{m}{n} = \frac{1,000,060,000}{1,200,060,000} = \frac{50,003}{60,003}$$

### Step-by-step continued fraction decomposition

$$\frac{50,003}{60,003} = \frac{1}{60,003/50,003} = \frac{1}{1 + \frac{10,000}{50,003}} \left( \approx \frac{1}{1} = 1 \right)$$

$$\frac{1}{1 + \frac{10,000}{50,003/10,000}} = \frac{1}{1 + \frac{1}{5 + \frac{3}{10,000}}} \left( \approx \frac{1}{1 + \frac{1}{5}} = \frac{5}{6} \right)$$

$$\frac{1}{1 + \frac{1}{5 + \frac{1}{10,000/3}}} = \frac{1}{1 + \frac{1}{5 + \frac{1}{3,333 + \frac{1}{3}}}}$$

$$\left( \approx \frac{1}{1 + \frac{1}{5 + \frac{1}{3,333}}} = \frac{16,666}{19,999} \right)$$

$$\frac{1}{1 + \frac{1}{5 + \frac{1}{3,333 + \frac{1}{3}}}} \quad \%(\text{exactly equals } 50,003/60,003)$$

Therefore the obtained continued fraction approximation terms are  $[0;1,5,3333,3]$ .

The approximations are:  $0, 0 + \frac{1}{1} = 1, 0 + \frac{1}{1 + \frac{1}{5}} = \frac{5}{6}, 0 + \frac{1}{1 + \frac{1}{5 + \frac{1}{3,333}}} = \frac{16,666}{19,999}, \dots$

Note that a decryption aware of the problem domain (like, magnitudes of numerator and denominator terms) would be able to exploit the big gap between the correct value and the next one, obtaining:

$$D(m, n) = \frac{5}{6}$$

Here this happens to be the exact value of the result, free of privacy-leaking redundant common factors. Note that in this application of continued fraction expansion, the criteria for estimation is based on the magnitudes of the values in numerator and denominator. Here 5, and 6 are closer to our problem domain values than  $\max(16666, 19999)$ .

## Continued Fraction Expansion Analysis

Note that our expansion was:

$$\frac{50,003}{60,003} = [0; 1, 5, 3333, 3]$$

### Convergents Table

Index	Continued Fraction	Value	Error	Term Magnitude
1	[0; 1]	$\frac{1}{1}$	0.16665833375	1
2	[0; 1, 5]	$\frac{5}{6}$	<b>0.0000833292</b>	1
3	[0; 1, 5, 3333]	$\frac{16666}{19999}$	0.00000000083	5
4	[0; 1, 5, 3333, 3]	$\frac{50003}{60003}$	0	5

Since  $\frac{5}{6}$  is within threshold and has a fitting magnitude for its terms (much smaller denominator than the next convergent)  $\frac{16666}{19999}$ , it is chosen. In general, a relatively big gap between magnitudes of approximations can be considered as a strong indication of reaching the exact result.

$$D(1,000,060,000, 1,200,060,000) \rightarrow \frac{5}{6}$$

### Experimental Evaluation

Having established approximate decryption as a post-decryption interpretation mechanism, we empirically evaluate masked rational division under two scaling strategies: **Shared- $k$**  and **Distinct** ( $k_a, k_b$ ). The evaluation focuses on numerical growth behavior and continued-fraction reconstruction complexity of the decrypted ratio. The evaluation focuses on numerical behavior and reconstruction characteristics under different scaling strategies. The results provide empirical insight into how implementation choices influence potential information leakage, without claiming formal security guarantees.

#### Rational Encoding and Encrypted Division

We consider exact rational inputs of the form

$$a = \frac{n_a}{d_a}, \quad b = \frac{n_b}{d_b}, \quad n_a, n_b \in \mathbb{Z}, d_a, d_b \in \mathbb{N}. \quad (10)$$

Encoding is applied at the plaintext level prior to encryption, and encrypted division employs a nonzero multiplicative term-hiding mask  $\varepsilon \neq 0$  to suppress deterministic post-decryption structure.

**Shared- $k$  Scaling:** Under the **Shared- $k$**  strategy, all (here both) operands are encoded using a common scaling factor

$$k = \text{lcm}(d_a, d_b) 10^t. \quad (11)$$

This is possible only when the secret dealers know both  $d_a$  and  $d_b$ . This would be relevant in cloud computations or in multi-party computations when evaluation of complex arithmetic circuits like  $a/c + b * d$  where  $a$  and  $b$  are distributed by the same owner and  $c$  and  $d$  by a different one.

The resulting fixed-point encodings are

$$(a_1, a_2) = (ak, k), \quad (b_1, b_2) = (bk, k). \quad (12)$$

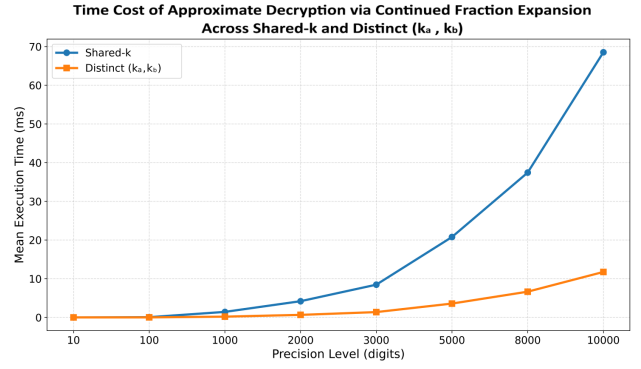


Figure 1: Execution time of approximate decryption under increasing precision.

Encrypted division then yields the masked ciphertext pair

$$(E(ak^2\varepsilon), E(bk^2\varepsilon)), \quad (13)$$

which induces quadratic numerical growth of order

$$\Theta(k^2). \quad (14)$$

**Distinct** ( $k_a, k_b$ ) **Scaling:** By contrast, the **Distinct** ( $k_a, k_b$ ) strategy assigns independent scaling factors

$$k_a = d_a 10^{t_a}, \quad k_b = d_b 10^{t_b}. \quad (15)$$

This can be used both in cloud applications and in multi-party secret computations.

The corresponding encodings are

$$(a_1, a_2) = (ak_a, k_a), \quad (b_1, b_2) = (bk_b, k_b). \quad (16)$$

Encrypted division in this case produces

$$(E(ak_a k_b \varepsilon), E(bk_a k_b \varepsilon)), \quad (17)$$

with numerical growth proportional to

$$\Theta(k_a k_b), \quad (18)$$

thereby reducing integer expansion relative to the Shared- $k$  formulation.

The parameters  $t$ ,  $t_a$ , and  $t_b$  control numerical precision only and are selected according to application-level accuracy requirements.

### Experimental Setup

Precision was varied across

$$\{10, 100, 1000, 3000, 5000, 8000, 10000\} \text{ digits.}$$

For each precision level, ten rational input instances were evaluated over 1,000 repetitions, yielding approximately  $10^4$  masked-division executions per data point. We record execution time, memory usage, and reconstruction error.

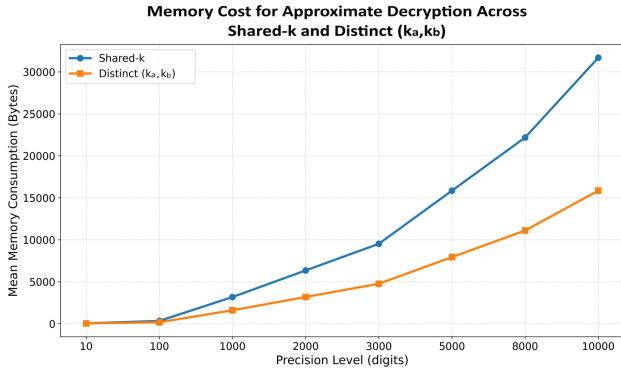


Figure 2: Memory consumption during continued-fraction reconstruction.

## Performance Results

Figures 1 and 2 illustrate the practical impact of the two masking strategies. As precision increases, Shared- $k$  incurs significantly higher computational and memory costs, consistent with its  $\Theta(k^2)$  growth. In contrast, Distinct  $(k_a, k_b)$  maintains smoother scaling due to reduced coupling between masked operands.

## Reconstruction Error

Table 1: Observed reconstruction error under approximate decryption

Method	Error Rate (%)
Shared- $k$	6.94
Distinct $(k_a, k_b)$	3.29

As shown in Table 1, Distinct  $(k_a, k_b)$  achieves lower reconstruction error than Shared- $k$ , making it preferable in accuracy-sensitive settings, whereas Shared- $k$  remains suitable when approximation tolerance is higher and faster, simpler computation is favored over reconstruction precision.

## Results and Discussion of Reconstruction Behavior

### Information Leakage Perspective

The output of a computation over secret inputs inevitably reveals some information about those inputs; this leakage is intrinsic to the semantics of the result and cannot be eliminated. Consider private inputs  $a = 0.6$  and  $b = 1.4$ , yielding the plaintext output  $\frac{a}{b} = \frac{0.6}{1.4} = \frac{3}{7}$ . This result unavoidably reveals that the denominator is a multiple of 7; moreover, if one operand is known (e.g.  $a = 6$ ), the other is uniquely determined ( $b = 14$ ). Such leakage is inherent to the output semantics and is therefore unavoidable under any correct computation model.

Our goal is to ensure that the *computation method does not leak more information than the output itself*, assuming

an adversary who observes the decrypted result and all public parameters, but does not know the secret inputs.

Using fixed-point precision  $k = 4$  ( $10^k = 10000$ ), we encode

$$(a_1, a_2) = (6000, 10000), \quad (b_1, b_2) = (14000, 10000).$$

To break deterministic structure prior to encrypted division, symmetric additive masking is applied:

$$E(a) = \frac{E(6003)}{E(10000)}, \quad E(b) = \frac{E(14003)}{E(10000)}.$$

Encrypted division yields

$$E(a/b) = \frac{E(6003)}{E(10000)} \cdot \frac{E(10000)}{E(14003)},$$

optionally followed by a nonzero multiplicative mask  $c = 2$ , producing

$$E(a/b) = \frac{E(120,060,000)}{E(280,060,000)}.$$

After decryption, the observer obtains

$$(m, n) = (120,060,000, 280,060,000) \Rightarrow \frac{m}{n} = \frac{6003}{14003}.$$

Applying continued fraction expansion,

$$\frac{6003}{14003} = [0; 2, 3, 166, \dots],$$

the early convergent  $\frac{3}{7}$  is recovered with absolute error  $\approx 5.1 \times 10^{-5}$ , which is within typical application-level tolerance.

**Security interpretation.** The recovered value  $\frac{3}{7}$  exposes only the semantic content of the output. In contrast, prior fixed-point schemes may reveal structured representations (e.g.,  $60000/140000$ ), enabling divisor-based inference on  $b$  through encoding artifacts. By suppressing this deterministic structure, the proposed masking-and-approximation approach limits the adversary’s inference strictly to what is implied by the output ratio itself; when  $a$  is unknown, no additional information about  $b$  is disclosed beyond consistency with the observed result. Using unpredictable values of  $C$ , different from  $10^k$ , has potentials to improve masking even further.

### Relationship to Random-Factor Masking

Random-factor masking techniques multiply both the numerator and denominator by a common factor  $c$ , preserving the ratio exactly after decryption. By contrast, additive masking introduces a distortion that survives decryption time; instead, the decrypted output can be further interpreted and corrected through continued fraction approximation, **recovering result without secret yielding redundant factors**. These approaches are therefore complementary rather than redundant. Multiplicative masking primarily mitigates direct invertibility while preserving exact ratios, whereas approximation-based masking targets inference channels where exact numerical recovery itself constitutes the leakage vector, through maintenance of redundant factors.

## Implications for Privacy-Preserving Machine Learning

The proposed approximation-based decryption framework aligns naturally with privacy-preserving machine learning and artificial intelligence systems that operate through aggregation or voting mechanisms rather than exact numerical recovery. In many distributed and ensemble-based ML settings—such as federated learning, secure model aggregation, and committee-based inference—individual models contribute partial numerical signals whose collective outcome depends on relative magnitude, ranking, or majority agreement rather than precise values (Fouque, Stern, and Wackers 2002). Within such workflows, approximate decryption provides a principled mechanism for extracting decision-relevant information while suppressing unnecessary numerical detail. By reconstructing low-denominator rational approximations via continued fractions, the scheme preserves ordering and proportional relationships sufficient for aggregation-based decisions, effectively functioning as a privacy-aware voting layer over encrypted outputs. This behavior is consistent with robustness properties widely exploited in ensemble learning and statistical learning theory, where bounded perturbations do not alter final predictions (Bonawitz et al. 2017; Corrigan-Gibbs and Boneh 2017). As a result, controlled approximation enables encrypted ML pipelines to reconcile privacy, scalability, and interpretability, allowing collective inference to proceed without exposing exact intermediate values or reintroducing structural invertibility.

### Limitations

The proposed method reveals a trade-off whereby tighter continued-fraction approximations improve numerical accuracy while increasing structural information exposure, whereas lower-denominator reconstructions reduce privacy leakage at the expense of precision, with parameters chosen according to application-level tolerance.

### Conclusion

Division-enabled homomorphic encodings based on rational representations can inadvertently permit operand recovery from decrypted division outputs, thereby undermining confidentiality through numerical invertibility (Chen 2017). Addressing this challenge, this work eliminates **deterministic invertibility** of redundant factors through the introduction of **symmetric additive masking** applied prior to encrypted division. Building on this foundation, the paper advances **approximate decryption** as a deliberate post-decryption interpretation mechanism, combining **continued-fraction reconstruction** with **precision thresholds** to recover application-acceptable results without exact operand disclosure. Furthermore, an **alternative masking instantiation** is presented, demonstrating a second numerical pathway for suppressing **structural linkage** before approximation. An extensive **empirical evaluation** under **Shared- $k$**  and **Distinct** ( $k_a, k_b$ ) scaling strategies characterizes numerical growth, reconstruction complexity, and approximation error, showing that **Distinct** ( $k_a, k_b$ )

achieves smoother scaling and lower reconstruction error. Taken together, these contributions position **controlled approximation** as a principled design dimension for managing **information exposure** in practical homomorphic division schemes. Besides, the approach aligns naturally with privacy-preserving machine learning and encrypted sensing scenarios, where bounded approximation error is acceptable (Boemer et al. 2019).

### References

- Acar, A.; Aksu, H.; Uluagac, A. S.; and Conti, M. 2018. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (Csur)* 51(4):1–35.
- Boemer, F.; Costache, A.; Cammarota, R.; and Wierzynski, C. 2019. ngraph-he2: A high-throughput framework for neural network inference on encrypted data. In *Proceedings of the 7th ACM workshop on encrypted computing & applied homomorphic cryptography*, 45–56.
- Bonawitz, K.; Ivanov, V.; Kreuter, B.; Marcedone, A.; McMahan, H. B.; Patel, S.; Ramage, D.; Segal, A.; and Seth, K. 2017. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 1175–1191.
- Brakerski, Z.; Gentry, C.; and Vaikuntanathan, V. 2014. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)* 6(3):1–36.
- Chen, H.; Han, K.; Huang, Z.; Jalali, A.; and Laine, K. 2017. Simple encrypted arithmetic library v2. 3.0. *Microsoft Research, December* 13.
- Chen, L. 2017. Cryptography standards in quantum time: new wine in old wineskin? *IEEE security & privacy* 15(4):51.
- Cheon, J. H.; Kim, A.; Kim, M.; and Song, Y. 2017. Homomorphic encryption for arithmetic of approximate numbers. In *International conference on the theory and application of cryptology and information security*, 409–437. Springer.
- Chung, H., and Kim, M. 2018. Encoding of rational numbers and their homomorphic computations for the-based applications. *International Journal of Foundations of Computer Science* 29(06):1023–1044.
- Corrigan-Gibbs, H., and Boneh, D. 2017. Prio: Private, robust, and scalable computation of aggregate statistics. In *14th USENIX symposium on networked systems design and implementation (NSDI 17)*, 259–282.
- ElGamal, T. 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory* 31(4):469–472.
- Fan, J., and Vercauteren, F. 2012. Somewhat practical fully homomorphic encryption. *Cryptology ePrint Archive*.
- Fouque, P.-A.; Stern, J.; and Wackers, G.-J. 2002. Cryptocomputing with rationals. In *International Conference on Financial Cryptography*, 136–146. Springer.

- Gentry, C. 2009. Fully homomorphic encryption using ideal lattices. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 169–178.
- Hardy, G. H., and Wright, E. M. 1979. *An introduction to the theory of numbers*. Oxford university press.
- Kelsey, J.; Schneier, B.; and Wagner, D. 1996. Key-schedule cryptanalysis of idea, g-des, gost, safer, and triple-des. In *Annual international cryptology conference*, 237–251. Springer.
- Khinchin, A. Y., and Teichmann, T. 1964. Continued fractions.
- Sidorov, V.; Wei, E. Y. F.; and Ng, W. K. 2022. Comprehensive performance analysis of homomorphic cryptosystems for practical data processing. *arXiv preprint arXiv:2202.02960*.
- Silaghi, M., and Alsulami, A. 2023. On enhancing security for division homomorphism with elgamal. In *The International FLAIRS Conference Proceedings*, volume 36.
- Yi, X.; Paulet, R.; Bertino, E.; Yi, X.; Paulet, R.; and Bertino, E. 2014. *Homomorphic encryption*. Springer.