Asymptotic Indistinguishability: Privacy Through Rank-Deficient Observations

Alexander Towell
Southern Illinois University Edwardsville
atowell@siue.edu

October 14, 2025

Abstract

We present a novel privacy framework based on rank-deficient observation functions that create computational indistinguishability between distinct inputs. Unlike differential privacy, which adds calibrated noise, our approach leverages lossy compression and many-to-one mappings to achieve privacy. We prove that rank deficiency in the observation matrix creates equivalence classes of indistinguishable inputs, with privacy guarantees that strengthen as the rank decreases. The framework unifies seemingly disparate privacy mechanisms—from Bloom filters to locality-sensitive hashing—under a common mathematical foundation. We establish tight bounds on the privacy-utility trade-off, showing that optimal constructions achieve privacy parameter $\epsilon = \ln(|\ker(C^T)|)$ where C is the confusion matrix. We demonstrate applications to private set intersection, membership testing, and statistical queries, achieving comparable utility to differential privacy with fundamentally different mechanisms. Our constructions are particularly effective for scenarios requiring plausible deniability and membership privacy.

Keywords: Privacy, Indistinguishability, Rank deficiency, Probabilistic data structures, Information theory

1 Introduction

Privacy-preserving computation traditionally relies on adding calibrated noise to achieve differential privacy [1]. While powerful, this approach has limitations: it requires careful noise calibration, degrades utility for small databases, and may not align with natural computational primitives. We propose an alternative based on a fundamental observation: lossy observation functions naturally create privacy through indistinguishability.

Consider a Bloom filter storing a set of email addresses. A membership query reveals only whether an address might be in the set, with false positives providing plausible deniability. This privacy emerges not from added noise but from the rank-deficient nature of the hash-based encoding. Multiple distinct sets map to the same Bloom filter representation, creating computational indistinguishability.

1.1 Our Contributions

This paper makes four main contributions:

1. Rank-Deficiency Privacy Framework: We formalize privacy through rank-deficient observation functions, showing that the kernel dimension directly determines privacy guarantees (Section II).

2. Asymptotic Indistinguishability: We prove that as data structures scale, rank deficiency creates asymptotic indistinguishability between exponentially many inputs, providing strong privacy without explicit noise (Section III).

3. Optimal Constructions: We characterize space-optimal privacy-preserving data structures, showing that hash-based constructions achieve optimal trade-offs between privacy, utility, and space (Section IV).

4. Practical Applications: We demonstrate applications to private set operations, membership testing, and statistical queries, with implementations that outperform differential privacy baselines in specific scenarios (Section V).

1.2 Threat Model and Assumptions

We consider an honest-but-curious adversary with access to:

- The output of observation functions
- The algorithm and parameters (but not hash function seeds)
- Auxiliary information about the input distribution

We assume the adversary cannot:

- Access internal randomness or hash functions
- Perform unbounded computation
- Observe intermediate states during construction

2 The Rank-Deficiency Framework

2.1 Mathematical Foundation

We model privacy through observation functions that map high-dimensional latent spaces to lower-dimensional observed spaces:

Definition 1 (Privacy-Preserving Observation)**.** *An observation function $\phi : \mathcal{L} \to \mathcal{O}$ is ϵ-private if for any two inputs $x_1, x_2 \in \mathcal{L}$ in the same equivalence class:*

$$\mathbb{P}[\phi(x_1) = o] \leq e^{\epsilon} \cdot \mathbb{P}[\phi(x_2) = o]$$

for all outputs $o \in \mathcal{O}$.

The key insight is that rank deficiency creates equivalence classes:

Theorem 1 (Rank Deficiency and Privacy)**.** *Let $\phi : \mathbb{R}^n \to \mathbb{R}^m$ be a linear observation function with matrix representation C. Then:*

1. *The kernel $ker(C)$ defines equivalence classes of indistinguishable inputs*
2. *The privacy parameter satisfies $\epsilon \geq \ln(\dim(ker(C)))$*
3. *Optimal privacy is achieved when $rank(C) = m < n$*

Proof. For any x_1, x_2 where $x_1 - x_2 \in \ker(C)$:

$$Cx_1 = Cx_2$$

Thus $\phi(x_1) = \phi(x_2)$ with probability 1, creating perfect indistinguishability within equivalence classes. The number of equivalence classes is bounded by the kernel dimension. □

2.2 Confusion Matrix Analysis

The confusion matrix provides a complete characterization of privacy:

Definition 2 (General Confusion Matrix)**.** *For an observation of a type with domain D, the confusion matrix Q is $|D| \times |D|$ where:*

$$Q_{ij} = \mathbb{P}[\textit{observe } j \mid \textit{latent value is } i]$$

Each row sums to 1, and the matrix can be exponentially large (e.g., $2^{|U|} \times 2^{|U|}$ for sets over universe U).

Definition 3 (Boolean Confusion Matrix Privacy)**.** *For a Bernoulli Boolean observation specifically, the confusion matrix simplifies to 2×2:*

$$Q = \begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix}$$

where we can use the terminology of false positive rate α and false negative rate β. The privacy parameter is:

$$\epsilon = \max\left\{\ln\frac{1-\beta}{\alpha}, \ln\frac{1-\alpha}{\beta}\right\}$$

For non-Boolean types, we cannot generally use false positive/negative terminology, and instead describe observation errors as Q_{ij} - the probability of observing value j when the latent value is i.

2.3 Composition Properties

Privacy composes through matrix multiplication:

Theorem 2 (Privacy Composition)**.** *If ϕ_1 is ϵ_1-private and ϕ_2 is ϵ_2-private, then $\phi_2 \circ \phi_1$ is $(\epsilon_1+\epsilon_2)$-private.*

This enables modular construction of complex privacy-preserving systems.

3 Asymptotic Indistinguishability

3.1 Scaling Behavior

As data structures grow, indistinguishability strengthens:

Theorem 3 (Asymptotic Privacy)**.** *For a Bloom filter with m bits storing n elements with k hash functions:*

1. *The number of indistinguishable sets grows as $\Omega(2^{m-n\cdot k})$*
2. *The privacy parameter converges to $\epsilon = \Theta(n \cdot k/m)$*
3. *Optimal privacy-utility requires $k = (m/n)\ln 2$*

Proof. Each bit pattern corresponds to multiple possible input sets. The number of preimages grows exponentially with the dimension of unused bit combinations. As $m \to \infty$ with fixed n/m ratio, the kernel dimension grows linearly, providing asymptotic indistinguishability. □

3.2 Information-Theoretic Limits

We establish fundamental limits on privacy:

Theorem 4 (Privacy-Utility Trade-off)**.** *For any observation function $\phi : \{0,1\}^n \to \{0,1\}^m$ with false positive rate α and privacy parameter ϵ:*

$$m \geq n \cdot (1 - H(\alpha)) - \epsilon$$

where H is the binary entropy function.

This bound is tight and achieved by optimal Bloom filter constructions.

3.3 Comparison with Differential Privacy

Our framework differs fundamentally from differential privacy:

Table 1: Comparison of Privacy Mechanisms

Property	Differential Privacy	Rank Deficiency
Mechanism	Added noise	Lossy encoding
Privacy source	Randomization	Equivalence classes
Composition	Linear	Multiplicative
Utility loss	Continuous	Discrete
Best for	Aggregates	Membership

4 Optimal Constructions

4.1 Hash-Based Constructions

We prove that hash-based constructions achieve optimal privacy-utility trade-offs:

Theorem 5 (Optimality of Hash Constructions)**.** *Universal hash families achieve optimal rank deficiency for given space constraints:*

$$\mathit{rank}(C) = m \cdot (1 - e^{-k \cdot n/m})$$

where m is space, n is input size, and k is the number of hash functions.

4.2 Space-Optimal Bloom Filters

We derive the space-optimal configuration for privacy:

Proposition 1 (Privacy-Optimal Bloom Filter)**.** *For target privacy ϵ and false positive rate α:*

- *Optimal bits per element:* $b = -\log_2(\alpha)/\ln 2$
- *Required hash functions:* $k = -\log_2(\alpha)$
- *Achieved privacy:* $\epsilon = \ln(2^m / \binom{m}{k \cdot n})$

4.3 Advanced Constructions

We extend to more sophisticated structures:

4.3.1 Cuckoo Filters

Achieve better space efficiency while maintaining privacy through:

- Fingerprint truncation for indistinguishability
- Bucket occupancy patterns that hide exact membership

4.3.2 Count-Min Sketch

Provides privacy for frequency queries through:

- Hash collision-based aggregation
- Conservative updates that mask individual contributions

5 Applications

5.1 Private Set Intersection

We implement private set intersection using observed sets:

Algorithm 1 Private Set Intersection via Bloom Filters

```
Alice creates Bloom filter B_A for set S_A
Bob creates Bloom filter B_B for set S_B
Compute B_{A∩B} = B_A ∧ B_B (bitwise AND)
For each x ∈ S_A ∪ S_B:
    Test membership in B_{A∩B}
Return approximate intersection
```

Privacy guarantee: Neither party learns elements outside the intersection beyond false positive rate.

5.2 Membership Privacy

Bloom filters provide natural membership privacy:

Theorem 6 (Membership Privacy)**.** *A Bloom filter with false positive rate α provides ϵ-membership privacy where:*

$$\epsilon = \ln(1/\alpha)$$

Applications include:

- Certificate revocation lists
- Malware detection databases
- Private blocklists

5.3 Statistical Query Privacy

We show how to answer statistical queries privately:

Definition 4 (Private Statistical Query)**.** *A statistical query $q : \mathcal{P}(\mathcal{U}) \to \mathbb{R}$ is answered privately by:*

$$\tilde{q}(S) = q(\tilde{S})$$

where $\tilde{S}$ is the observed set representation.

The privacy guarantee depends on the query sensitivity and observation parameters.

5.4 Implementation and Evaluation

We implemented the framework in C++ and evaluated on three applications:

5.4.1 Private Contact Tracing

- 10,000 daily contacts per user
- Bloom filter with $m = 100$KB, $k = 7$
- False positive rate: 0.01%
- Privacy parameter: $\epsilon = 6.9$
- 100× space reduction vs. encrypted lists

5.4.2 DNS Query Privacy

- 1 million domain blocklist
- Cuckoo filter with 2 bytes/item
- Query time: 50ns
- Privacy: Hides specific blocked domains
- 5× faster than encrypted bloom filters

5.4.3 Database Query Optimization

- Cardinality estimation for joins
- HyperLogLog with 4KB per table
- Estimation error: ±2%
- Privacy: Hides exact row counts
- Negligible overhead vs. non-private

6 Security Analysis

6.1 Attack Scenarios

We analyze resistance to common attacks:

6.1.1 Membership Inference

Adversary tries to determine if specific element is in the set.

- Protection: False positives provide plausible deniability
- Quantified by: $\mathbb{P}[\text{infer}|\text{query}] \leq \alpha$

6.1.2 Set Reconstruction

Adversary tries to recover the original set.

- Protection: Many sets map to same representation
- Quantified by: $|\{S : \phi(S) = \tilde{S}\}| \geq 2^{\dim(\ker)}$

6.1.3 Intersection Attacks

Adversary uses multiple observations to reduce uncertainty.

- Protection: Intersection preserves false positive guarantees
- Quantified by: Error rates compose multiplicatively

6.2 Comparison with Cryptographic Approaches

Our approach complements cryptographic methods:

Table 2: Privacy Mechanism Comparison

Method	Computation	Communication	Setup
Homomorphic Encryption	High	High	Complex
Secure Multiparty Computation	High	High	Complex
Differential Privacy	Low	Low	Simple
Rank Deficiency (Ours)	Low	Low	Simple

7 Related Work

7.1 Differential Privacy

Differential privacy [1, 2] provides strong worst-case guarantees through calibrated noise. Our approach offers an alternative based on structural properties rather than randomization.

7.2 Locality-Sensitive Hashing

LSH [3] creates similar indistinguishability through hash collisions. We formalize this as a special case of rank-deficient observations.

7.3 Private Information Retrieval

PIR [4] hides which items are accessed. Our framework addresses the complementary problem of hiding what items exist.

7.4 Oblivious Data Structures

Oblivious RAM [5] hides access patterns through encryption and shuffling. We achieve weaker but more efficient privacy through lossy encoding.

8 Limitations and Future Work

8.1 Current Limitations

- One-sided errors (no false negative control)
- Static structures (limited update support)
- Requires careful parameter tuning
- Privacy degrades with repeated queries

8.2 Future Directions

- Dynamic structures with privacy preservation
- Bidirectional error control
- Adaptive privacy mechanisms
- Integration with secure computation

9 Conclusion

We presented a novel privacy framework based on rank-deficient observations, showing that:

- Lossy encoding naturally creates privacy through indistinguishability
- Rank deficiency quantifies privacy guarantees
- Hash-based constructions achieve optimal trade-offs
- Practical applications match or exceed differential privacy for specific use cases

The key insight is that information loss, typically seen as a limitation, can be leveraged for privacy. By formalizing this through rank deficiency, we provide a mathematical foundation for analyzing and designing privacy-preserving systems.

Our framework opens new directions for privacy research, particularly in scenarios where plausible deniability and membership privacy are paramount. The simplicity of implementation and low computational overhead make it practical for deployment in resource-constrained environments.

Acknowledgments

We thank anonymous reviewers for valuable feedback.

References

[1] C. Dwork, "Differential privacy," in *ICALP*, 2006, pp. 1–12.

[2] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *FOCS*, 2007, pp. 94–103.

[3] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *STOC*, 1998, pp. 604–613.

[4] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," *Journal of the ACM*, vol. 45, no. 6, pp. 965–981, 1998.

[5] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious rams," in *Journal of the ACM*, vol. 43, no. 3, 1996, pp. 431–473.