Binomial DP Noise Generation in MPC

IETF drafts

1. Simple and Efficient Binomial Protocols for Differential Privacy in MPC
2. Efficient Protocols for Binary Fields in the 3-Party Honest Majority MPC Setting
3. High Performance Pseudorandom Secret Sharing (PRSS)

Outline

1. DP Noise in MPC
   a. Motivation
   b. Binomial Noise in MPC
   c. Binomial DP-Mechanism
   d. Setting parameters
   e. Performance cost
2. 3-Party Honest Majority MPC
3. PRSS

Motivation

Why generate DP noise in MPC?

1. Better privacy/utility trade off compared to each Worker adding DP noise
2. Noise is not significantly impacted by the size of the query such as with local, shuffle DP noise.

Why use Binomial noise?

1. Simple to generate in MPC (flip coins, sum).
2. Meaningful improvement over baselines for many common parameter sets.

Binomial noise in MPC

Bin(N,p) is the number of successes in N Bernoulli p trials.

To sample in MPC, two things are needed:

1. A protocol for Bernoulli trials, or coin-flipping protocol, that produces a value of 1 with probability p and 0 otherwise.
2. A means to sum the value of N trials.

Binomial DP Mechanism

Function computed on a private dataset: $f(D) \to \mathbb{Z}$

In MPC

1. Generate a sample X from a Bin(N,p) distribution
2. Output = f(D) + X

Recipient computes

1. Output - N*p for a unbiased result

Binomial DP Mechanism

Function computed on a private dataset: $f(D) \rightarrow \mathbb{Z}^d$

In MPC

1. Generate samples X in $\mathbb{Z}^d$ from a Bin(N,p) distribution
2. Output = f(D) / s + X for s=1 / j, j a natural number

Recipient computes

1. s * (output - N*p) for a unbiased, unscaled result

Binomial DP Mechanism

Function computed on a private dataset: f(D) -> $\mathbb{Z}^d$

In MPC

1. Generate samples X in $\mathbb{Z}^d$ from a Bin(N,p) distribution
2. Output = f(D) * j + X for j a natural number

Recipient computes

1. (output - N*p) / j for a unbiased, unscaled result

Setting Bin(N,p) to get Approximate-DP guarantee

For a given (ε,δ), how to set parameters of the Bin(N,p) distribution?

See 2018 paper [CPSGD]

1. p = 0.5 is optimal, which is also the easiest for a coin-flip protocol in MPC.
2. For determining N from (ε,δ) there are two constraints:
   a. Delta only constraint
   b. Epsilon and delta constraint
3. Calculate the smallest N such that both constraints are simultaneously satisfied.

DP constraints on N

delta_constraint

- N >= 4·max(23·ln(10·d/delta), 2·sensitivity_infty/s)

epsilon_delta_constraint

- epsilon = sensitivity_2·sqrt(2·ln(1.25/delta)) / (s/2·sqrt(N)) + (sensitivity_2·c□·sqrt(ln(10/delta)) + sensitivity_1·b□) / ((s/4)·(1-delta/10) · N) + (2/3·sensitivity_infty·ln(1.25/delta) + sensitivity_infty·d□·ln(20·d/delta)·ln(10/delta)) / ((s/4)·N)

Can solve as a quadratic equation in N, or simply binary search to find smallest N satisfying both.

Setting Quantization Scale

Output = f(D) / s + X

Decreasing the quantization scale reduces the error added but increases N (MPC cost).

Decrease s as much as possible subject to practical MPC cost constraints (s being 1/integer, N being manageably small).

Comparison of error with ideal Gaussian in MPC

Ideal error for a single Gaussian perfectly generated in MPC would be

- d * (2 * ell_2_sensitivity ^2 * ln(1.25 / delta)) / epsilon^2

Error for Binomial in MPC is

- d * s^2 Np (1-p)

Error for independent Gaussians added by Helpers:

- With three helpers would be 50% more than ideal.
- With two parties it would be twice the ideal (100% more).

Table of Select Parameters (N < 1M)

epsilon	delta	sensitivity	dimension	s	N	% error is worse than ideal
0.1	10^{-9}	1	32	0.3	292k	57.1%
1	10^{-8}	32	1	1	227k	48.8%
1	10^{-8}	32	100	1	241k	57.7%
1	10^{-7}	32	1	1	193k	44.9%
5	10^{-7}	32	1	0.1	660k	23.3%
5	10^{-6}	32	100	0.1	581k	26.3%

Table of Select Parameters (N < 10M)

epsilon	delta	sensitivity	dimension	s	N	% error is worse than ideal
0.01	10^{-9}	1	1	1	1.9M	17.8%
0.01	10^{-9}	1	1	0.5	7.3M	9.1%
0.1	10^{-8}	1	1	0.05	6.4M	8.4%
0.1	10^{-8}	16	1	1	4.2M	10.5%
1	10^{-7}	16	1	0.1	3.6M	9.5%
5	10^{-7}	1	1	0.005	285k	36.4%
5	10^{-7}	1	1	0.001	5.6M	7.7%
5	10^{-7}	64	1	0.05	9M	6%

Table of Select Parameters (N < 100M)

epsilon	delta	sensitivity	dimension	s	N	% error is worse than ideal
0.01	10^{-9}	1	32	0.2	44M	4.1%
0.01	10^{-9}	1	32	0.15	76M	3.1%
0.1	10^{-7}	64	32	1	55M	2.8%
0.1	10^{-7}	1	32	0.01	133M	1.8%
1	10^{-7}	32	32	0.05	55M	2.8%
5	10^{-7}	32	32	0.008	86M	2.2%
5	10^{-7}	16	32	0.01	14M	5.6%
5	10^{-7}	16	32	0.005	55M	2.8%

Summary of parameters

- To improve upon the two party baseline (100% -> 50% worse than ideal), you need N to be approximately [200k, 300k]
- To improve upon the three party baseline (50% -> 10% worse than ideal), you need N to be approximately [1M , 10M]
- To improve further to 2% - 3% worse than ideal you need N [10M, 100M]

MPC cost to generate Bin(N,p) in 3 party honest majority

For **256** samples generated in parallel (vectorized) the bandwidth and latency scales as follows with increasing N.

N	Bandwidth	Latency
1M	95 MB	~3 min
10M	959 MB	~34 min
100M	9,590 MB*	~340 min*

*extrapolated

Ongoing or Future Work

1. We are working to improve the analysis for small epsilon, large sensitivity parameters. (e.g. increasing the quantization scale instead of decreasing)
2. We are working on a new three party aggregation which would decrease the latency
3. Instantiations of sampling in the two-party setting which would allow this to be used in the DAP setting.
4. See if other approaches to central DP in MPC can improve over Binomials

3-Party MPC & PRSS

Multiplications for boolean circuits in 3-party MPC

- Semi-honest multiplication requires **1 bit** of communication per helper
  - One AES encryption per helper (shared randomness)
- Active security:
  - Implementation is based on distributed zero-knowledge proofs.
  - Validation is done in batches of **50 M** multiplications:
    - **6 Mb** of network bandwidth per helper
    - **100 MB** RAM to store the intermediates

Pseudorandom Secret Sharing (PRSS)

Draft: [High Performance Pseudorandom Secret Sharing]

Method for sampling shared secret randomness.

Building block of MPC protocols.

IETF drafts

1. Simple and Efficient Binomial Protocols for Differential Privacy in MPC
2. Efficient Protocols for Binary Fields in the 3-Party Honest Majority MPC Setting
3. High Performance Pseudorandom Secret Sharing (PRSS)