Online Differentially Private Consistent Clustering

Edith Cohen* **Vadym Doroshenko†** **Badih Ghazi‡** **Pritish Kamath§** **Alexander Knop§**

Ravi Kumar§ **Ethan Leeman§** **Pasin Manurangsi§** **Adam Sealfon§** **Marika Swanberg§**

Abstract

We study differentially private (DP) k-means and k-median clustering in the online streaming setting. In this model, points arrive sequentially, and at each time step, we need to output a set of k centers that optimizes the clustering objective for all points seen so far. We give a generic reduction that transforms the (sensitive) input stream into a private stream, which is a *semi-coreset* of the input stream. This implies that any (non-private) online clustering algorithm, run as a post-processing step, can achieve good utility for the original clustering objective. Our algorithm matches or improves upon the approximation ratio, space usage, and running time of existing algorithms [Epasto et al., 2026, Dupré la Tour et al., 2024]. A key aspect of our reduction is that it inherits desirable properties of the underlying non-private clustering algorithm, such as *consistency* [Lattanzi and Vassilvitskii, 2017]—a property not satisfied by previous DP algorithms.

1 Introduction

Differential Privacy (DP) [Dwork et al., 2006a,b] has emerged as the mathematically rigorous, widely used gold standard for privacy-preserving large-scale data analysis. By injecting carefully calibrated statistical noise into computations, DP provides a formal, quantifiable guarantee that limits the information an adversary can infer about any single data point, making it a foundational tool for government agencies and technology companies alike (e.g., Abowd [2018], Erlingsson et al. [2014]).

Within the domain of large-scale data analysis, clustering remains one of the most basic unsupervised learning tasks. In this work, we focus on the (k,p)-*clustering objective*: given a dataset $X = (x_1,\ldots,x_n) \in (\mathbb{R}^d)^n$, the goal is to output a set C of k centers that minimizes $\mathrm{cost}_X^p(C) := \sum_{i\in[n]} \mathrm{dist}(x_i,C)^p$, where $\mathrm{dist}(x_i,C)$ denotes the (Euclidean) distance from x_i to its closest point in C. When $p=1$ (resp., 2), this corresponds to the k-*median* (resp., k-*means*) clustering, which are among the most well-studied objectives. Since finding the exact minimizer is NP-hard, clustering algorithms focus on providing a small multiplicative *approximation ratio* [Matousek, 2000, Kanungo et al., 2004, Har-Peled and Mazumdar, 2004, Awasthi et al., 2015, Lee et al., 2017, Cohen-Addad and Karthik C. S., 2019, Friggstad et al., 2019, Ahmadian et al., 2020]. We assume throughout that $p \geq 1$ is a fixed constant and we will omit the dependency on p in notations and bounds for brevity.

Traditionally, clustering algorithms were designed for static datasets, where $x_1,\ldots,x_n$ are processed simultaneously. However, in modern applications, continuous data generation and collection are prevalent. Online clustering is a natural task that arises in analyzing dynamically changing data streams. In the streaming (a.k.a. continuous release) model, the data points $x_1,\ldots,x_n$ arrive sequentially and after the arrival of each x_t, the algorithm must return a set C_t of k centers for all

*Google Research and Tel Aviv University. edith@cohenwang.com

†Google. dvadym@google.com

‡Google Deepmind. badihghazi@gmail.com

§Google Research. pritish@alum.mit.edu, alexanderknop@google.com, ravi.k53@gmail.com, ethanleeman@google.com, pasin@google.com, adamsealfon@google.com, marikaswanberg@google.com

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

the data points seen so far, i.e., $x_1, \ldots, x_t$. Non-private algorithms with small approximation ratios and low space complexity are known in this model, e.g., [Guha et al., 2000, Charikar et al., 2003, Har-Peled and Mazumdar, 2004].

DP Clustering. Given the significance of clustering, a vast body of literature has explored DP clustering algorithms [Blum et al., 2005, Gupta et al., 2010, Balcan et al., 2017, Feldman et al., 2017, Nissim et al., 2016, Nissim and Stemmer, 2018, Huang and Liu, 2018, Stemmer, 2020, Ghazi et al., 2020, Chang et al., 2021, Ghazi et al., 2023, Dupré la Tour et al., 2024]. In particular, significant progress has been made in the static setting, where efficient DP clustering algorithms with constant approximation ratios and nearly-optimal additive errors are known [Ghazi et al., 2020, Dupré la Tour et al., 2024]. Unfortunately, the same cannot be said in the streaming model. Specifically, while Dupré la Tour et al. [2024] proposed a DP online (k, p)-clustering algorithm with nearly optimal approximation ratios and additive errors, their algorithm is inefficient, requiring super-polynomial time and space. Meanwhile, Epasto et al. [2026] proposed an efficient algorithm—based on the merge-and-reduce framework for coresets [Agarwal et al., 2004, Feldman et al., 2020]—but with suboptimal additive errors in terms of the dimension d. Furthermore, neither algorithm is designed to preserve desirable properties (such as consistency) of their underlying non-private counterpart, as explained next.

Consistency in Online Clustering. While online clustering algorithms must adapt to changing data, this can cause a volatile output sequence. For example, if an algorithm blindly recalculates cluster centers based on $x_1, \ldots, x_t$ for every $t \in [n]$, the cluster centers can vary wildly even across consecutive time steps. The need for a temporally stable output sequence motivated the definition of consistency [Lattanzi and Vassilvitskii, 2017]: An output sequence $(C_1, \ldots, C_n)$ is said to be *m-consistent* if $\sum_{t\in[n]} |C_t \setminus C_{t-1}| \le m$ where we use the convention $C_0 = \emptyset$, i.e., the total number of centers removed/added throughout the entire sequence is at most $m = m(n)$. Consistency ensures that the clusters do not change unpredictably at every time step. Maintaining consistency is preferred because temporal instability can be undesirable in many real-world scenarios.

Since the notion was formalized, many consistent clustering algorithms have been proposed in the literature [Lattanzi and Vassilvitskii, 2017, Guo et al., 2021, Fichtenberger et al., 2021, Chan et al., 2025]. In particular, Fichtenberger et al. [2021] devised a non-private algorithm for clustering in arbitrary metric space that achieves $O(1)$-approximation and $\tilde{O}(k)$-consistency (which is known to be nearly optimal [Lattanzi and Vassilvitskii, 2017]). Unfortunately, as mentioned earlier, neither of the previous DP online clustering algorithms [Dupré la Tour et al., 2024, Epasto et al., 2026] provides non-trivial guarantees on the consistency of their output sequence.

1.1 Our Contributions

Our primary contribution is a novel generic reduction that transforms a sensitive stream to a privatized stream that approximately preserves the clustering objective value on any input prefix. As a key corollary, we obtain the first DP algorithm that has a nearly optimal consistency guarantee. Furthermore, our algorithm obtains a constant approximation ratio, nearly optimal additive error in terms of the dimension d, and almost linear time and sublinear space for $d, k = n^{o(1)}$, as stated below.

Theorem 1 (Informal version of Corollary 11)**.** *There is an (ε, δ)-DP online (k, p)-clustering algorithm that, given a stream $X = (x_1, \ldots, x_n)$ of points, outputs a stream $C = (C_1, \ldots, C_n)$ of center multisets such that for each $t \in [n]$, $|C_t| = k$ and with high probability, C_t is an approximate solution for $x_1, \ldots, x_t$ with constant approximation ratio and $\frac{\sqrt{d}}{\varepsilon} \cdot (k \log(nd/\delta))^{O(1)}$-additive error. Moreover, the output is $(k \cdot \operatorname{polylog}(nd))$-consistent.*

Furthermore, when $k, d, \log(1/\delta) \le n^{o(1)}$, the algorithm runs in $n^{1+o(1)}$ time over the entire stream and uses $n^{o(1)}$ space.

As mentioned above, both the dependency of $\sqrt{d} \cdot \operatorname{polylog}(d)$ in the additive error, and the consistency bound $k \cdot \operatorname{polylog}(nd)$ are nearly optimal, matching the known lower bounds [Peter et al., 2024, Lattanzi and Vassilvitskii, 2017] up to polylogarithmic factors. A detailed comparison between our algorithm and those in Dupré la Tour et al. [2024], Epasto et al. [2026] can be found in Table 1.

Table 1: Comparison of online (ε, δ)-DP clustering algorithms. For brevity, we hide $\mathrm{polylog}\,(nd/\delta)$ terms in all bounds and assume $\varepsilon = \Theta(1)$. All algorithms have constant approximation ratio. While Dupré la Tour et al. [2024] do not specify the complexity of their algorithm, the bottleneck in their algorithm is the use of an $O(1/n)$-net in $\mathbb{R}^{O(\log k)}$, which requires $n^{\Theta(\log k)}$ time and space. We note however that their algorithm supports stream with both insertions and deletions, whereas Epasto et al. [2026] and ours only support insertions. Note that $o_{k,d}(n)$ consistency cannot be achieved for streams with both insertions and deletions even non-privately and when $O_{k,d}(1)$ additive error is allowed.

Algorithm	Additive Err.	Space	Time	Consistency
Dupré la Tour et al. [2024]	$k^{O(1)}\sqrt{d}$	$n^{O(\log k)}d^{O(1)}$	$n^{O(\log k)}d^{O(1)}$	—
Epasto et al. [2026]	$k^{2.5}d^{3.51}$	$(kd)^{O(1)}$	$n(dk)^{O(1)}$	—
This paper (Corollary 11)	$k^{2.5}\sqrt{d}$	$(kd)^{O(1)}$	$ndk^2 + (dk)^{O(1)}$	k

We now state our generic reduction. Recall that for a given multiset X, a semi-coreset $\tilde{X}$ is its "compressed" version such that for any set of cluster centers, the clustering objective on X and $\tilde{X}$ are close to each other, up to a multiplicative error in the approximation factor and an additive error in the cost (see Definition 4.) Our reduction is an online algorithm that produces an output stream $\tilde{X} = (\tilde{x}_1, \ldots, \tilde{x}_n)$ of points[5] such that, for any $t \in [n]$, the output prefix $\tilde{x}_1, \ldots, \tilde{x}_t$ is a semi-coreset of the input prefix $x_1, \ldots, x_t$, as stated below; furthermore, our algorithm is DP.

Theorem 2 (Informal version of Theorem 9)**.** *There is an (ε, δ)-DP algorithm that, given a stream $X = (x_1, \ldots, x_n)$, produces another stream $\tilde{X} = (\tilde{x}_1, \ldots, \tilde{x}_n)$ such that, with high probability, $\{\tilde{x}_1, \ldots, \tilde{x}_t\}$ is a semi-coreset for $\{x_1, \ldots, x_t\}$ with constant multiplicative error and $\frac{\sqrt{d}}{\varepsilon} \cdot (k \log(nd/\delta))^{O(1)}$-additive error for all $t \in [n]$.*

Furthermore, when $k, d, \log(1/\delta) \le n^{o(1)}$, the algorithm runs in $n^{1+o(1)}$ time over the entire stream and uses $n^{o(1)}$ space.

Since DP is preserved under post-processing, we can run any *non-private* online clustering algorithm on the output $\tilde{X}$. Indeed, our DP consistent clustering algorithm (Corollary 11) simply runs the (non-private) algorithm by Fichtenberger et al. [2021] as a post-processing step on $\tilde{X}$.

1.2 Technical Overview

(Non-Private) Meyerson's Sketch. The starting point of our work is the so-called *Meyerson's sketch* [Meyerson, 2001]. To describe the sketch, it is simpler to consider the setting where, instead of keeping a semi-coreset, we only aim for an *end-stream bi-criteria* approximation. Here, we need to output a set S of points at the end of the input stream $X = (x_1, \ldots, x_n)$ such that (i) $\mathrm{cost}^p_X(S)$ is at most $O(1)$ times the optimum and (ii) $|S| \le O(k \log n)$. We use the nomenclature "bi-criteria" since we allow S to be (slightly) larger than k. Standard techniques exist for turning such an end-stream bi-criteria approximation into an end-stream semi-coreset, even with DP; see §C for details.

We now describe the Meyerson sketch. Initially, $S = \emptyset$. After a point x_t arrives, we compute its distance to its closest point in S, denoted by $\mathrm{dist}(x_t, S)$. Then, with probability λ_t, we add it to S; otherwise, we simply discard it. Here, λ_t is set based on $\mathrm{dist}(x_t, S)$ so that the larger the distance, the more likely it is to be included in S. The full description is in § 1.2, which also uses an estimate (i.e., "guess") of the optimum value $\mathrm{opt}_{\mathrm{est}}$. Standard analysis shows that if $\mathrm{opt}_{\mathrm{est}}$ is within a constant factor of the true optimum, then the output S is a bi-criteria approximation with constant probability [Meyerson, 2001, Lattanzi and Vassilvitskii, 2017, Fichtenberger et al., 2021].

Algorithm 1: Meyerson's Sketch [Meyerson, 2001]

Require: Stream $X = (x_1, \ldots, x_n)$
Parameters: Estimated optimum $\mathrm{opt}_{\mathrm{est}}$
$S = \emptyset$
for $t = 1, \ldots, n$ **do**
  $\lambda_t \leftarrow \min\left\{1, \frac{k \log n}{\mathrm{opt}_{\mathrm{est}}} \cdot \mathrm{dist}(x_t, S)^p\right\}$
  {If $S = \emptyset$, let $\mathrm{dist}(x_t, S) = \infty$}
  Sample $Z_t \sim \mathrm{Ber}(\lambda_t)$
  if $Z_t = 1$ **then**
    $S \leftarrow S \cup \{x_t\}$
Output S

[5] As stated our algorithm outputs a stream of *multisets* of points rather than individual points, i.e., multiple points may be released in each step. See Theorem 9 for the formal statement. We note that it is possible to modify the reduction so that at most one point is released in each step, but this distinction is not important in downstream applications.

Privacy via DP Dense Balls. A naive attempt to make Meyerson's sketch (in fact, any sampling-based sketch) DP faces the following issue: The output contains (raw) input points, rendering it blatantly non-private. To overcome this challenge, we do *not* return the set S directly. Instead, we maintain another set R, which serves as a proxy for S but is DP, and use the distance $\mathrm{dist}(x_t, R)$ instead of $\mathrm{dist}(x_t, S)$ in each step. Our main observation is that the analysis of the Meyerson's sketch is quite flexible: as long as R contains a "representative point" in every "dense ball" of S, it is already sufficient to obtain a bi-criteria approximation (with some small additive error) while still ensuring that S remains not "too large". We formulate this requirement for the proxy set R as the *dense ball* problem (Definition 7) and show that known algorithms from the literature [Nissim et al., 2016, Nissim and Stemmer, 2018, Ghazi et al., 2020] can be adapted to solve this problem privately.

With this formulation, we then compute R by running a DP dense ball algorithm on S. However, doing so at every time step $t \in [n]$ would mean we need to apply the DP composition theorem on the privacy budget across all the steps, resulting in excessive error (that is polynomial in n). To overcome this, we observe that since S remains small, it suffices to invoke the DP dense ball algorithm only when S is updated. However, we cannot naively perform this check since S contains sensitive data; hence, we use the *continual counting* mechanism [Dwork et al., 2010, Chan et al., 2011] to provide a DP estimate of $|S|$ after every time step. We then invoke the DP dense ball algorithm on S only when the (privately estimated) size of S is sufficiently larger than what it was when the DP dense ball was last invoked. This concludes our high-level technical overview.

Sampling is the most popular method for finding a (semi-)coreset for (non-private) clustering problems and beyond (e.g., Chen [2009], Feldman and Langberg [2011], Feldman et al. [2020], Cohen-Addad et al. [2021]). We believe our work is the first to naturally integrate sampling-based algorithms with DP, and hope that our techniques will be useful for future research on this topic.

2 Preliminaries

For any metric space $\mathcal{M} = (U, \mathrm{dist})$, let $\mathcal{B}(u, r)$ denote the *ball* of radius r centered at u, i.e., $\mathcal{B}(u, r) := \{u' \in U \mid \mathrm{dist}(u, u') \le r\}$. For any $S \subseteq U$, let $\mathrm{dist}(u, S) = \inf_{u' \in S} \mathrm{dist}(u, u')$.

Throughout this work, we assume that we work with the Euclidean space[6] with unit diameter[7].

Assumption 3. *$\mathcal{M}$ is the Euclidean space in $\mathbb{R}^d$ with $U = \mathcal{B}\left(0, \frac{1}{2}\right)$.*

A *stream* (of points) is a sequence $X = (x_1, \ldots, x_n)$, where each x_t belongs to $U \cup \{\perp\}$. Here $\perp$ should be interpreted as "no point"; for convenience, we abuse the notation and write $\mathrm{dist}(\perp, S) = 0$ for any set $S \subseteq U$. For every $t \in [n]$, we write $X_{\le t}$ to denote the multiset $\{x_i \mid i \in [t], x_i \neq \perp\}$.

We also define a *multiset stream* $Y = (y_1, \ldots, y_n)$, where each y_i is a multiset of U. Again, we write $Y_{\le t}$ to denote the multiset $\bigcup_{i \in [t]} y_i$. The *size* of Y is $|Y_{\le n}|$.

An *online algorithm* $\mathcal{A}$ is an algorithm that receives the points in the stream $X = (x_1, \ldots, x_n)$ one point per time step, and, immediately after receiving x_t at time step t, it must output o_t. The final output released over the execution is $\mathcal{A}(\mathcal{X}) = (o_1, \ldots, o_n)$.

2.1 (Online) (k, p)-Clustering

Static Clustering. For a given multiset $X = \{x_1, \ldots, x_n\} \subseteq U$ of input points, the cost for a set $C \subseteq U$ of centers is given by $\mathrm{cost}^p_X(C) := \sum_{i \in [n]} \mathrm{dist}(x_i, C)^p$, where $p \ge 1$ is a constant. In the (k, p)-*clustering problem*, the goal is to find a set C of k centers that minimizes $\mathrm{cost}^p_X(C)$. It will be convenient to define the optimum as $\mathrm{opt}^{k,p}_X := \min_{C \in \binom{U}{k}} \mathrm{cost}^p_X(C)$.

Online Clustering. For a given stream $X = (x_1, \ldots, x_n)$, a multiset stream $C = (C_1, \ldots, C_n)$ is an (α, β)-*approximation* if

$$\mathrm{cost}^p_{X_{\le t}}(C_t) \le \alpha \cdot \mathrm{opt}^{k,p}_{X_{\le t}} + \beta \qquad \forall t \in [n].$$

[6] Our DP Meyerson's sketch (Algorithm 2) works for any metric space as long as there is an efficient DP dense ball algorithm for that space. However, we are only aware of existing work for the Euclidean space.

[7] The bounded diameter assumption is necessary to obtain finite additive error [Nissim et al., 2016]. By scaling, our algorithm works for any diameter bound Δ, incurring a multiplicative factor of Δ^p in the additive error.

An *online* (k,p)*-clustering algorithm* takes in an input stream $X = (x_1, \ldots, x_n)$ and outputs a multiset stream $C = (C_1, \ldots, C_n)$ such that $|C_t| = k$ for all $t \in [n]$. We say that it is an (α, β)-*approximation* algorithm for online k-clustering if for every input stream, the output stream of the algorithm is an (α, β)-approximation (with probability $1 - \gamma$).

Prefix-Approximation. For our Meyerson's sketch construction, it will be helpful to define the following notion. We say that a multiset stream $Y = (y_1, \ldots, y_n)$ is an (α, β)-*prefix-approximation* (w.r.t. (k,p)-clustering) of a stream $X = (x_1, \ldots, x_n)$ if

$$\sum_{t \in [n]} \mathrm{dist}(x_t, Y_{\le t})^p \le \alpha \cdot \mathrm{opt}^{k,p}_{X_{\le n}} + \beta.$$

Note that prefix-approximation is stronger than requiring $Y_{\le n}$ to be an (α, β)-approximation of $X_{\le n}$, since prefix-approximation computes the cost using $\mathrm{dist}(x_t, Y_{\le t})$. In other words, we only allow x_t to be connected to the point closest to it only up until that point in the stream (or, equivalently, the points in $y_{t+1}, \ldots, y_n$ cannot be used). On the other hand, we do not restrict the size of Y (which will be stated explicitly in the theorem statements, e.g., Theorem 31).

2.1.1 Semi-Coreset

Next, we recall the definition of a *semi-coreset* [Epasto et al., 2026]. Unlike a standard coreset [Har-Peled and Mazumdar, 2004], which provides a purely multiplicative approximation, a semi-coreset allows additional additive error terms.

Definition 4 (Semi-Coreset [Epasto et al., 2026])**.** *A multiset $\tilde{X}$ is a (κ, η_1, η_2)-*semi-coreset *for (k,p)-clustering of X if, for all $C \in \binom{U}{k}$, we have*

$$\tfrac{1}{\kappa} \cdot \mathrm{cost}^p_X(C) - \eta_1 \cdot \mathrm{opt}^{k,p}_X - \eta_2 \;\le\; \mathrm{cost}^p_{\tilde{X}}(C) \;\le\; \kappa \cdot \mathrm{cost}^p_X(C) + \eta_2.$$

When k, p are clear from context, we often drop them from the notation for brevity.

We say that a multiset stream $\tilde{X} = (\tilde{x}_1, \ldots, \tilde{x}_n)$ is a (κ, η_1, η_2)-semi-coreset for another multiset stream $X = (x_1, \ldots, x_n)$ if $\tilde{X}_{\le t}$ is a (κ, η_1, η_2)-semi-coreset of $X_{\le t}$ for all $t \in [n]$.

While the semi-coreset definition is more relaxed than the original coreset definition, it still maintains the most crucial property of coreset: Any good approximation for the semi-coreset is also a good approximation for the original dataset. This is stated more precisely below[8].

Lemma 5 (Approximation from Semi-Coreset)**.** *Let X be a multiset of points and let $\tilde{X}$ be a (κ, η_1, η_2)-semi-coreset for X. If a set $C \in \binom{U}{k}$ of centers is an (α, β)-approximation for $\tilde{X}$, then C is an (α', β')-approximation for X, where $\alpha' = \alpha\kappa^2 + \eta_1\kappa$ and $\beta' = \kappa(\alpha + 1)\eta_2 + \kappa\beta$.*

2.2 Consistent Clustering

For an online (k,p)-clustering algorithm, we say that an output stream $C = (C_1, \ldots, C_n)$ of centers is m-*consistent* if $\sum_{t \in [n]} |C_t \setminus C_{t-1}| \le m$, where $m = m(n)$. We say that the algorithm is m-consistent if its output is always m-consistent. We use the following result by Fichtenberger et al. [2021], which yields a nearly optimal consistency bound in the non-private setting.

Theorem 6 (Consistent k-Clustering with Constant Additive Error [Fichtenberger et al., 2021])**.** *There is an online (k,p)-clustering $(k \cdot \mathrm{polylog}\,(nd))$-consistent algorithm whose output is an $(O(1), O(1))$-approximation with high probability. Moreover, its time and space complexity is polynomial in the number of distinct points in the input stream and the number of dimensions d.*

We remark that the above result is slightly different than the one originally stated by Fichtenberger et al. [2021], whose bound depends on the "aspect ratio" of the metric space. However, there is a simple reduction between the two versions. See § A.2 for more details.

[8] Missing proofs in this section (including Lemma 5) and additional preliminaries can be found in Appendix A.

2.3 Dense Balls

As mentioned in § 1.2, a central problem in our algorithm is the dense ball problem, which is parameterized by $r_{\min}$, the smallest target radius, and τ, the target number of points per ball. The input to the problem is a multiset S of points and the output is another multiset O. (Let $\digamma = 3$ be a factor that is fixed throughout.) The problem is formalized as follows.

Definition 7 (Dense Ball)**.** *We say that an algorithm solves the* $(r_{\min}, \tau)$-dense ball *(with probability* $1-\gamma$*) if the following* coverage constraint *holds for any input multiset* S *of at most* N *points (with probability* $1-\gamma$*):* $\operatorname{dist}(u, O) \le \digamma \cdot r$ *for all* $u \in U, r \ge r_{\min}$ *such that* $|S \cap \mathcal{B}(u, r)| \ge \tau$.

Moreover, we say that the algorithm has size blow-up Υ with size capacity N *if, for any input* S *of size at most* N*, the output* O *(always) satisfies* $|O| \le \Upsilon \cdot \frac{N}{\tau}$.

2.4 DP and Continual Counting

We quickly recall the definition of DP. For online clustering algorithms, two input streams $X = (x_1, \ldots, x_n), X' = (x'_1, \ldots, x'_n)$ are *(add-remove) neighbors*[9] iff there exists $i^* \in [n]$ such that $x_i = x'_i$ for all $i \neq i^*$, and $\perp \in \{x_{i^*}, x'_{i^*}\}$. In words, one stream can be obtained from the other by changing a single point to $\perp$.

Definition 8 (Differential Privacy (DP) [Dwork et al., 2006a,b])**.** *An online algorithm* $\mathcal{A}$ *that takes in a stream is said to be* (ε, δ)-DP *if, for all sets* S *of possible outputs and all pairs* X, X' *of neighboring input streams, it holds that* $\Pr[\mathcal{A}(X) \in S] \le e^{\varepsilon} \cdot \Pr[\mathcal{A}(X') \in S] + \delta$.

The case $\delta = 0$ is known as *pure-DP* (written as ε-DP) and $\delta > 0$ is known as *approximate-DP*.

Throughout this work, we will assume that $\varepsilon \in (0, 1)$ and $\delta \in (0, 1/2)$.

We remark that the above model is the so-called *non-adaptive* (aka *oblivious*) model where the adversary fixes the neighboring input streams at the start of the algorithm. Our privacy proof however works even with the stronger *adaptive* (aka *adversarial*) model where the adversary may choose to adapt the streams based on the outputs so far. Due to the extra notations required, we defer the formal definition of this adaptive model to the appendix. However, we stress that our utility proof assumes the oblivious model, and it remains an interesting question to extend the utility guarantees to the adversarial setting. (See §5 for more discussion.)

Continual Counting. In DP continual counting [Dwork et al., 2010, Chan et al., 2011], the input stream is $Y = (y_1, \ldots, y_n) \in \{0, 1\}^n$. The goal is to output a private estimate e_t of $\sum_{t' \in [t]} y_{t'}$ after receiving y_t. We say that the estimate is α-*accurate* if $\left| e_t - \sum_{t' \in [t]} y_{t'} \right| \le \alpha$ for all $t \in [n]$

3 Private Semi-Coreset Construction

Our main DP online semi-coreset construction (a formal version of Theorem 2) is stated below.

Theorem 9 (DP Online Semi-Coreset)**.** *There is an online* (ε, δ)*-DP algorithm whose output is an* $(O(1), O(1), \beta)$*-semi-coreset of the input with probability* $1-\gamma$*, and the algorithm runs in time* $\tilde{O}\left(ndk^2 \cdot \log(1/\gamma)\right) + \Lambda^{O(1)}$ *and uses space* $\Lambda^{O(1)}$ *where*

- $\beta = O\left(\frac{d \cdot k^3}{\varepsilon} \cdot \operatorname{polylog}\left(\frac{nd}{\varepsilon\gamma}\right)\right)$ *and* $\Lambda = \frac{kd \log(n/\gamma)}{\varepsilon}$ *for pure-DP.*
- $\beta = O\left(\frac{\sqrt{d} \cdot k^{2.5}}{\varepsilon} \cdot \operatorname{polylog}\left(\frac{nd}{\varepsilon\delta\gamma}\right)\right)$ *and* $\Lambda = \frac{kd \log\left(\frac{n}{\gamma\delta}\right)}{\varepsilon}$ *for approximate-DP.*

3.1 From Dense Ball to Prefix-Approximation: Private Meyerson's Sketch

The main ingredient for Theorem 9 is a private version of Meyerson's sketch (§ 1.2). As in the non-private version, we maintain a set S_t of sampled input points, but instead of using these directly we periodically feed them to the DP dense ball algorithm to get R_t, which we actually output. For

[9] Note that one may consider the *substitution neighbors*, in which we can change x_{i^*} to any other point x'_{i^*} (not necessarily $\perp$). By standard *group privacy*, our algorithm also applies in this setting.

Algorithm 2 Private Meyerson's Sketch: $\mathcal{A}_{\text{Meyerson}}(\text{opt}_{\text{est}}, \Psi)$

Require: Stream $X = (x_1, \ldots, x_n)$
Parameters: Target sketch size Ψ; Dense ball algorithm $\mathcal{A}_{\text{dense}}(\tau, \Psi, \Upsilon)$ with threshold τ, size capacity $\Psi \cdot \tau$ and size blow-up Υ; Continual counting algorithm $\mathcal{A}_{\text{count}}$; Estimated optimum opt_{est}

1: $(R_0, S_0) \leftarrow (\{0\}, \emptyset)$
2: $j = 1$ {Counter for when to call $\mathcal{A}_{\text{dense}}$}
3: **for** $t = 1, \ldots, n$ **do**
4: {Do the following upon receiving x_t}
5: $\lambda_t \leftarrow \min\left\{1, \frac{\tau k L}{\text{opt}_{\text{est}}} \cdot \text{dist}(x_t, R_{\le t-1})^p\right\}$ {Probability of adding x_t to S_t}
6: $S_t \leftarrow S_{t-1}$
7: Sample $Z_t \sim \text{Ber}(\lambda_t)$ {Should x_t be added to S_t?}
8: **if** $Z_t = 1$ and $|S_t| < \Psi \cdot \tau$ **then**
9: $S_t \leftarrow S_t \cup \{x_t\}$
10: Feed Z_t to $\mathcal{A}_{\text{count}}$ and let e_t be the output
11: **if** $e_t \ge j \cdot \tau$ and $j \le \Psi$ **then**
12: $R_t \leftarrow \mathcal{A}_{\text{dense}}(S_t)$
13: $j \leftarrow j + 1$
14: **else**
15: $R_t \leftarrow \emptyset$
16: Output R_t

notational convenience, let $R_0 = \{0\}$ contain the origin, and write $R_{\le t} = \bigcup_{i=0}^{t} R_i$ to include R_0 as well. Moreover, we use the convention $\text{dist}(\perp, R_{\le t}) = 0$ so that $x_t = \perp$ is never included in the set S_t. Recall that opt_{est} is a guess of the optimum value. Let $r_{\min} := \left(\frac{\text{opt}_{\text{est}}}{n}\right)^{1/p}$ and $L := \left\lceil \log_2\left(\frac{1}{r_{\min}}\right)\right\rceil$. Algorithm 2 follows the outline from § 1.2 except for additional checks to ensure that the size of S_t and the number of calls to the DP dense ball algorithm are bounded.

A generic version of the approximation guarantee of Algorithm 2 is given below.

Lemma 10 (Prefix-Approximation Guarantee)**.** *Assume that*

- $\mathcal{A}_{\text{dense}}(\tau, \Psi, \Upsilon)$ *solves* $(r_{\min}, \tau)$*-dense ball and has size blow-up* Υ *with size capacity* $\Psi \cdot \tau$*.*
- $\mathcal{A}_{\text{count}}$ *solves continual counting with accuracy* $\tau/2$*.*
- $\Psi \ge D \cdot \left(kL \cdot \left(1 + \frac{\text{opt}^{k,p}_{X_{\le n}}}{\text{opt}_{\text{est}}}\right)\right)$ *for some sufficiently large constant* D*.*

Then, the output stream $(R_1, \ldots, R_n)$ *is an* $(O(1), O(\text{opt}_{\text{est}} + \tau k L))$*-prefix-approximation of* $(x_1, \ldots, x_n)$ *with probability* $1/2$*. Furthermore, the output stream has size* $|R_{\le n}| \le \Psi^2 \cdot \Upsilon$*.*

Proof. The size bound immediately follows due to the checks $j \le \Psi$ on line 11, $|S_t| \le \Psi \cdot \tau$ on line 8 and the size blow-up guarantee of $\mathcal{A}_{\text{dense}}$.

Next, we will prove the prefix-approximation claim. To do this, it will be more convenient to analyze a modified version of Algorithm 2, denoted Algorithm 2′, removing the checks that $|S_t| < \Psi \cdot \tau$ on line 8 and that $j \le \Psi$ on line 11. For Algorithm 2′, we will then show the following:

$$\mathbb{E}\left[\sum_{t\in[n]} \text{dist}(x_t, R_{\le t-1})^p\right] \le O\left(\text{opt}^{k,p}_{X_{\le n}} + \text{opt}_{\text{est}} + \tau k L\right), \tag{1}$$

$$\mathbb{E}\left[|S_{\le n}|\right] \le \tfrac{\Psi\cdot\tau}{8}. \tag{2}$$

Before we prove these, we show how they complete our main proof. By Markov's inequality and a union bound, (1), (2) imply that, w.p. $3/4$, both the prefix-approximation guarantee and $|S_t| \le \Psi \cdot \tau$ holds in Algorithm 2′. When the latter holds (and since $\mathcal{A}_{\text{count}}$ is $(\tau/2)$-accurate), both checks on lines 8 and 11 in Algorithm 2 pass; this means that Algorithm 2 produces the same output $(R_1, \ldots, R_t)$ as Algorithm 2′. Thus, the prefix-approximation guarantee holds with probability $3/4$ for Algorithm 2.

To prove (1) and (2), let $C^* = \{c_1^*, \ldots, c_k^*\}$ denote the optimal centers such that $\text{cost}^p_{X_{\le n}}(C^*) = \text{opt}^{k,p}_{X_{\le n}}$. For each $i \in [k]$, let $P_i = \{t \in [n] \mid x_t \text{ is closest to } c_i^*, \text{ ties broken arbitrarily }\}$.

For every $\ell = 1, \dots, L$, define the partitions

$$P_{i,\ell} := \begin{cases} \{t \in P_i \mid \mathrm{dist}(x_t, c_i^*) \le 2 \cdot r_{\min}\} & \text{if } \ell = 1, \\ \{t \in P_i \mid 2^{\ell-1} \cdot r_{\min} \le \mathrm{dist}(x_t, c_i^*) \le 2^{\ell} \cdot r_{\min}\} & \text{otherwise.} \end{cases}$$

We note that $\bigcup_{\ell \in [L]} P_{i,\ell} = P_i$, since the diameter of $\mathcal{M}$ is at most one.

Consider each $P_{i,\ell}$. Referring back to Algorithm 2, recall that Z_t is the outcome of the coin, determined by λ_t, that decides if x_t should be added to S_t. For every $j \in P_{i,\ell}$, we define $Z_{<j} := \sum_{\substack{j' \in P_{i,\ell} \\ j' < j}} Z_{j'}$, i.e., the number of elements in this partition that has been added until before time step j. Similarly, we define $\lambda_{<j} := \sum_{\substack{j' \in P_{i,\ell} \\ j' < j}} \lambda_{j'}$.

Proof of (1). Let us define additional notation:

$$\widetilde{Z}_t = \begin{cases} Z_t & \text{if } Z_{<t} < 3\tau, \\ 0 & \text{otherwise.} \end{cases} \qquad \widetilde{\lambda}_t = \begin{cases} \lambda_t & \text{if } Z_{<t} < 3\tau, \\ 0 & \text{otherwise.} \end{cases}$$

Consider any $t \in P_{i,\ell}$. Observe that, if $Z_{<t} \ge 3\tau$ (i.e., $\lambda_t \ne \widetilde{\lambda}_t$), then $|S_{t-1} \cap \mathcal{B}(c_i^*, 2^\ell \cdot r_{\min})| \ge 3\tau$. Let $t' < t$ be the last time step $\mathcal{A}_{\mathrm{dense}}$ is called before time step t. Due to the guarantee of $\mathcal{A}_{\mathrm{count}}$, we must have $|S_t \setminus S_{t'}| \le \tau + 2 \cdot (\tau/2) = 2\tau$. This implies that $Z_{<t'} \ge \tau$. By the guarantee of $\mathcal{A}_{\mathrm{dense}}$, there must be a point $f \in R_{\le t-1}$ such that $\mathrm{dist}(c_i^*, f) \le O(2^\ell \cdot r_{\min})$. This implies

$$\begin{aligned} \mathrm{dist}(x_t, R_{\le t-1}) \le \mathrm{dist}(x_t, c_i^*) + \mathrm{dist}(c_i^*, f) &\le \mathrm{dist}(x_t, c_i^*) + O(2^\ell \cdot r_{\min}) \\ &\le O(\mathrm{dist}(x_t, c_i^*) + \mathbf{1}[\ell = 1] \cdot r_{\min}). \end{aligned} \tag{3}$$

Meanwhile, by definition of λ_t and since the diameter of $\mathcal{M}$ is at most one, we also have that

$$\mathrm{dist}(x_t, R_{\le t-1})^p \le \lambda_t \cdot \left(1 + \tfrac{\mathrm{opt}_{\mathrm{est}}}{\tau k L}\right). \tag{4}$$

Using (3) when $\lambda_t \ne \widetilde{\lambda}_t$ and (4) when $\lambda_t = \widetilde{\lambda}_t$, we have

$$\mathrm{dist}(x_t, R_{\le t-1})^p \le O\left(\mathrm{dist}(x_t, c_i^*)^p + \mathbf{1}[\ell = 1] \cdot r_{\min}^p + \widetilde{\lambda}_t \cdot \left(1 + \frac{\mathrm{opt}_{\mathrm{est}}}{\tau k L}\right)\right).$$

Taking expectation, we thus get

$$\mathbb{E}\left[\mathrm{dist}(x_t, R_{\le t-1})^p\right] \le O\left(\mathrm{dist}(x_t, c_i^*)^p + \mathbf{1}[\ell = 1] \cdot r_{\min}^p + \mathbb{E}\left[\widetilde{Z}_t\right] \cdot \left(1 + \frac{\mathrm{opt}_{\mathrm{est}}}{\tau k L}\right)\right).$$

Summing over all $t \in [n]$, we get

$$\mathbb{E}\left[\textstyle\sum_{t \in [n]} \mathrm{dist}(x_t, R_{\le t-1})^p\right] \le O\left(\mathrm{opt}_{X_{\le n}}^{k,p} + n \cdot r_{\min}^p + \mathbb{E}\left[\textstyle\sum_{t \in [n]} \widetilde{Z}_t\right] \cdot \left(1 + \tfrac{\mathrm{opt}_{\mathrm{est}}}{\tau k L}\right)\right).$$

Finally, by definition, $\sum_{t \in P_{i,\ell}} \widetilde{Z}_t \le 3\tau$ regardless of the randomness of the algorithm. Summing over $i \in [k]$ and $\ell \in [L]$ gives $\sum_{t \in [n]} \widetilde{Z}_t \le 3\tau k L$. Plugging this into the above gives

$$\begin{aligned} \mathbb{E}\left[\textstyle\sum_{t \in [n]} \mathrm{dist}(x_t, R_{\le t-1})^p\right] &\le O\left(\mathrm{opt}_{X_{\le n}}^{k,p} + n \cdot r_{\min}^p + \tau k L + \mathrm{opt}_{\mathrm{est}}\right) \\ &\le O\left(\mathrm{opt}_{X_{\le n}}^{k,p} + \mathrm{opt}_{\mathrm{est}} + \tau k L\right), \end{aligned}$$

where the last inequality is due to our choice of $r_{\min}$. This proves (1).

Proof of (2). First, recall that (3) holds if $Z_{<t} \ge 3\tau$, i.e., $\widetilde{Z}_t \ne Z_t$. In this case, we have

$$\mathbb{E}\left[Z_t - \widetilde{Z}_t\right] = \mathbb{E}\left[Z_t\right] = \lambda_t \overset{(3)}{\le} O\left(\frac{\tau k L \cdot (\mathrm{dist}(x_t, c_i^*)^p + \mathbf{1}[\ell = 1] \cdot r_{\min}^p)}{\mathrm{opt}_{\mathrm{est}}}\right).$$

Thus, the expected total number of points sampled from $P_{i,\ell}$ is

$$\mathbb{E}\left[\textstyle\sum_{t \in P_{i,\ell}} Z_t\right] = \mathbb{E}\left[\textstyle\sum_{t \in P_{i,\ell}} \widetilde{Z}_t\right] + \mathbb{E}\left[\textstyle\sum_{t \in P_{i,\ell}} (Z_t - \widetilde{Z}_t)\right]$$

$$\leq 3\tau + O\left(\sum_{t\in P_{i,\ell}} \frac{\tau kL\cdot(\mathrm{dist}(x_t,c_i^*)^p + \mathbf{1}[\ell=1]\cdot r_{\min}^p)}{\mathrm{opt}_{\mathrm{est}}}\right),$$

where in the first inequality we again use the fact that $\sum_{t\in P_{i,\ell}} \widetilde{Z}_t \leq 3\tau$ by definition.

Summing this up over all $i \in [k]$ and $\ell \in [L]$, we get

$$\mathbb{E}\left[|S_{\leq n}|\right] \leq O\left(\tau kL + \frac{\tau kL\cdot\left(\mathrm{opt}^{k,p}_{X_{\leq n}} + n\cdot r_{\min}^p\right)}{\mathrm{opt}_{\mathrm{est}}}\right) \leq O\left(\tau kL + \frac{\tau kL\cdot \mathrm{opt}^{k,p}_{X_{\leq n}}}{\mathrm{opt}_{\mathrm{est}}}\right),$$

where the second inequality is due to our choice of $r_{\min}$. Finally, the RHS is at most $\frac{\Psi\cdot\tau}{8}$ for sufficiently large constant D. Thus, (2) holds as desired. □

The privacy and efficiency analysis of the sketch is deferred to Appendix B.1. Finally, turning such a prefix-approximation into a coreset can be easily done via snapping each point x_t to the closest center in the output so far. Again, we need use a continual counting algorithm to compute the weights in DP manner. This is a standard technique that has been applied before in the DP setting (e.g., Ghazi et al. [2020]). The full argument is deferred to Appendix C.

4 From DP Semi-Coreset to DP Consistent Clustering

As mentioned earlier, we can simply invoke the non-private online consistent clustering algorithm (Theorem 6) on the private semi-coreset (Theorem 9) to obtain the desired result.

Corollary 11. *There is an (ε,δ)-DP online (k,p)-clustering $(k\cdot\mathrm{polylog}\,(nd))$-consistent algorithm whose output is an $(O(1),\beta)$-approximation with high probability, and the algorithm runs in time $\tilde{O}\left(ndk^2\right) + \Lambda^{O(1)}$ and uses space $\Lambda^{O(1)}$ where*

- $\beta = O\left(\frac{d\cdot k^3}{\varepsilon}\cdot\mathrm{polylog}\left(\frac{nd}{\varepsilon}\right)\right)$ *and* $\Lambda = \frac{kd\log n}{\varepsilon}$ *for pure-DP.*
- $\beta = O\left(\frac{\sqrt{d}\cdot k^{2.5}}{\varepsilon}\cdot\mathrm{polylog}\left(\frac{nd}{\varepsilon\delta}\right)\right)$ *and* $\Lambda = \frac{kd\log(n/\delta)}{\varepsilon}$ *for approximate-DP.*

Proof. We run the (ε,δ)-DP semi-coreset algorithm from Theorem 9 on input stream X to get a multiset stream $\tilde{X} = (\tilde{x}_1,\ldots,\tilde{x}_n)$. We then feed the points in $\tilde{x}_1,\ldots,\tilde{x}_n$ to the online consistent (k,p)-clustering algorithm in Theorem 6; the final output C_t is the output from this algorithm after feeding it $\tilde{x}_t$. The consistency claim and time/space bounds immediately follow. The privacy guarantee holds because of the post-processing property of DP, as we only post-process $(\tilde{x}_1,\ldots,\tilde{x}_n)$.

Finally, since C_t is an $(O(1),O(1))$-approximation of $\tilde{X}_{\leq t}$ and $\tilde{X}_{\leq t}$ is an $(O(1),O(1),\beta)$-semi-coreset of $X_{\leq t}$, Lemma 5 implies that C_t is also an $(O(1),O(\beta))$-approximation of $X_{\leq t}$. □

5 Conclusion and Open Questions

In this work, we presented a generic reduction that transforms a sensitive data stream into a privatized semi-coreset stream. By allowing any non-private online clustering algorithm to be executed as a post-processing step, our framework yields the first DP online clustering algorithm that achieves a nearly optimal consistency guarantee, alongside a constant approximation ratio and a nearly optimal additive error in terms of the dimension.

Despite these advances, several open questions and directions for future work remain:

- **Practicality**: While our private adaptation of Meyerson's sketch (Algorithm 2) is straightforward to implement and practical, the underlying DP dense ball algorithms it relies upon are theoretical. To our knowledge, there are currently no practical implementations of these algorithms, which poses a barrier to real-world deployment.
- **Error Dependency on k:** While our additive error dependency on dimension d is nearly optimal, the dependency on k leaves room for improvement. In our private Meyerson's sketch (Algorithm 2), this sub-optimality is partly due to DP composition across all runs

of the dense ball algorithm; it might be possible to improve on this by deriving an "online" version of DP dense ball. More generally, while efficient DP algorithms with tight additive errors in terms of k are known in the offline setting [Nguyen et al., 2021, Ghazi et al., 2024], none is known for the online setting.

- **Coreset with $(1 \pm o(1))$-Multiplicative Error:** Our online construction (and that of Epasto et al. [2026]) only yields a semi-coreset, which naturally leaves the question of whether an online DP coreset construction (i.e., obtaining $\eta_1 = 0$ in Definition 4) is possible. Another interesting question is whether one can reduce the multiplicative error κ to $1 + o(1)$. We remark that the latter question is still open even in the offline setting, although there is some evidence that this might be hard [Ghazi et al., 2026].
- **Adversarial Streaming Models:** While our privacy analysis holds for the adversarial setting (where the adversary sees outputs up to time $i - 1$ before selecting x_i), our current utility analysis assumes an oblivious data stream. There are several challenges. For instance, it has not yet been established whether the non-private Meyerson's sketch maintains its utility guarantee in the adversarial model. In fact, we are not aware of any analysis of the (non-private) consistent k-means/median clustering in this model.
- **Running Time Optimization:** Although our algorithm achieves $n^{1+o(1)}$ time complexity over the entire stream for $k, d \leq n^{o(1)}$, the running time could be optimized further, e.g., by employing approximate nearest neighbor data structures. Furthermore, it also seems plausible that the Λ term in our main theorem (Theorem 9) can be optimized further. Currently, the polynomial dependency on k, d through Λ is due to the use of the 1-Cluster algorithm by Ghazi et al. [2020] as a blackbox (see Appendix A.4). It remains an interesting question whether this can be sped up further, e.g., to nearly linear time in k.
- **Deterministic Alternatives to Sampling:** It is possible to modify our private Meyerson's sketch (Algorithm 2) to avoid random sampling: instead of sampling each point x_t with probability λ_t, one can keep all points and assign x_t a weight of λ_t, adapting the DP continual counting and DP dense ball subroutines to this weighted setting. While this yields similar privacy and utility guarantees, it requires storing many low-weight points, resulting in polynomial space complexity. Indeed, even in the non-private setting, we are not aware of any consistent clustering algorithm with low space complexity that does not employ sampling, making this an interesting direction for future work.

AI Disclosure. While all the main algorithms and proofs in this work were discovered by the authors, we used AI, including Gemini (Google) and Claude (Anthropic), to assist in preparing this paper. In particular, we used AI to flesh out proof sketches to first drafts of full proofs, and to help with drafting the introduction and conclusion. These drafts are heavily edited and verified by the authors, and we take full responsibility for all claims in this work.

References

John M. Abowd. The U.S. Census Bureau adopts differential privacy. In *KDD*, pages 6353–6356, 2018.

Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan. Approximating extent measures of points. *J. ACM*, 51(4):606–635, 2004.

Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. Better guarantees for k-means and Euclidean k-median by primal-dual algorithms. *SIAM J. Comput.*, 49(4), 2020.

Pranjal Awasthi, Moses Charikar, Ravishankar Krishnaswamy, and Ali Kemal Sinop. The hardness of approximation of Euclidean k-means. In *SoCG*, pages 754–767, 2015.

Maria-Florina Balcan, Travis Dick, Yingyu Liang, Wenlong Mou, and Hongyang Zhang. Differentially private clustering in high-dimensional Euclidean spaces. In *ICML*, pages 322–331, 2017.

Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the SuLQ framework. In *PODS*, pages 128–138, 2005.

T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26:1–26:24, 2011.

T.-H. Hubert Chan, Shaofeng H.-C. Jiang, Tianyi Wu, and Mengshi Zhao. Online clustering with nearly optimal consistency. In *ICLR*, 2025.

Alisa Chang, Badih Ghazi, Ravi Kumar, and Pasin Manurangsi. Locally private k-means in one round. In *ICML*, pages 1441–1451, 2021.

Moses Charikar, Liadan O'Callaghan, and Rina Panigrahy. Better streaming algorithms for clustering problems. In *STOC*, pages 30–39, 2003.

Ke Chen. On coresets for k-median and k-means clustering in metric and Euclidean spaces and their applications. *SICOMP*, 39(3):923–947, 2009.

Vincent Cohen-Addad and Karthik C. S. Inapproximability of clustering in L_p metrics. In *FOCS*, pages 519–539, 2019.

Vincent Cohen-Addad, David Saulpic, and Chris Schwiegelshohn. A new coreset framework for clustering. In *STOC*, pages 169–182, 2021.

Damien Desfontaines and Balázs Pejó. SoK: Differential privacies. *PETS*, 2020(2):288–313, 2020.

Max Dupré la Tour, Monika Henzinger, and David Saulpic. Making old things new: A unified algorithm for differentially private clustering. In *ICML*, 2024.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, pages 486–503, 2006a.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006b.

Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *STOC*, pages 715–724, 2010.

Alessandro Epasto, Tamalika Mukherjee, and Peilin Zhong. Differentially private clustering in data streams. In *AISTATS*, 2026.

Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *CCS*, pages 1054–1067, 2014.

Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *STOC*, pages 569–578, 2011.

Dan Feldman, Chongyuan Xiang, Ruihao Zhu, and Daniela Rus. Coresets for differentially private k-means clustering and applications to privacy in mobile sensor networks. In *IPSN*, pages 3–16, 2017.

Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, PCA, and projective clustering. *SIAM J. Comput.*, 49(3):601–657, 2020.

Hendrik Fichtenberger, Silvio Lattanzi, Ashkan Norouzi-Fard, and Ola Svensson. Consistent k-clustering for general metrics. In *SODA*, pages 2660–2678, 2021.

Zachary Friggstad, Mohsen Rezapour, and Mohammad R. Salavatipour. Local search yields a PTAS for k-means in doubling metrics. *SIAM J. Comput.*, 48(2):452–480, 2019.

Badih Ghazi, Ravi Kumar, and Pasin Manurangsi. Differentially private clustering: Tight approximation ratios. In *NeurIPS*, 2020.

Badih Ghazi, Junfeng He, Kai Kohlhoff, Ravi Kumar, Pasin Manurangsi, Vidhya Navalpakkam, and Nachiappan Valliappan. Differentially private heatmaps. In *AAAI*, pages 7696–7704, 2023.

Badih Ghazi, Pritish Kamath, Ravi Kumar, Pasin Manurangsi, and Adam Sealfon. Individualized privacy accounting via subsampling with applications in combinatorial optimization. In *ICML*, 2024.

Badih Ghazi, Cristóbal Guzmán, Pritish Kamath, Alexander Knop, Ravi Kumar, and Pasin Manurangsi. Computational hardness of private coreset. In *FORC*, 2026.

Sudipto Guha, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering data streams. In *FOCS*, pages 359–366, 2000.

Xiangyu Guo, Janardhan Kulkarni, Shi Li, and Jiayi Xian. Consistent k-median: Simpler, better and robust. In *AISTATS*, pages 1135–1143, 2021.

Anupam Gupta, Katrina Ligett, Frank McSherry, Aaron Roth, and Kunal Talwar. Differentially private combinatorial optimization. In *SODA*, pages 1106–1125, 2010.

Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *STOC*, pages 291–300, 2004.

Monika Henzinger, Roodabeh Safavi, and Salil P. Vadhan. Concurrent composition for differentially private continual mechanisms. *SIGMOD*, 4(2):99:1–99:26, 2026.

Zhiyi Huang and Jinyan Liu. Optimal differentially private algorithms for k-means clustering. In *PODS*, pages 395–408, 2018.

Palak Jain, Sofya Raskhodnikova, Satchit Sivakumar, and Adam D. Smith. The price of differential privacy under continual observation. In *ICML*, pages 14654–14678, 2023.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. *Comput. Geom.*, 28(2-3):89–112, 2004.

Silvio Lattanzi and Sergei Vassilvitskii. Consistent k-clustering. In *ICML*, pages 1975–1984, 2017.

Euiwoong Lee, Melanie Schmidt, and John Wright. Improved and simplified inapproximability for k-means. *Inf. Process. Lett.*, 120:40–43, 2017.

Konstantin Makarychev, Yury Makarychev, and Ilya P. Razenshteyn. Performance of Johnson-Lindenstrauss transform for k-means and k-medians clustering. *SIAM J. Comput.*, 52:269–297, 2019.

Jirí Matousek. On approximate geometric k-clustering. *Discret. Comput. Geom.*, 24(1):61–84, 2000.

Adam Meyerson. Online facility location. In *FOCS*, pages 426–431, 2001.

Gaspard Monge. *Mémoire sur la théorie des déblais et des remblais*. De l'Imprimerie Royale, 1781.

Huy L. Nguyen, Anamay Chaturvedi, and Eric Z. Xu. Differentially private k-means via exponential mechanism and max cover. In *AAAI*, pages 9101–9108, 2021.

Kobbi Nissim and Uri Stemmer. Clustering algorithms for the centralized and local models. In *ALT*, pages 619–653, 2018.

Kobbi Nissim, Uri Stemmer, and Salil P. Vadhan. Locating a small cluster privately. In *PODS*, pages 413–427, 2016.

Naty Peter, Eliad Tsfadia, and Jonathan R. Ullman. Smooth lower bounds for differentially private algorithms via padding-and-permuting fingerprinting codes. In *COLT*, pages 4207–4239, 2024.

Uri Stemmer. Locally private k-means clustering. In *SODA*, pages 548–559, 2020.

Abhradeep Guha Thakurta and Adam D. Smith. (Nearly) optimal algorithms for private online learning in full-information and bandit settings. In *NeurIPS*, 2013.

A Additional Background and Missing Proofs from §2

When we state the running time and space complexity of an algorithm, we always assume that each coordinate of x_i fits in a single word of memory and arithmetic operations on them take constant time. This, in particular, means that reading each x_i takes $O(d)$ time and storing each x_i uses $O(d)$ space.

Approximate Triangle Inequality. The following "approximate triangle inequality" for the pth power will be useful in our proofs.

Lemma 12 (Approximate Triangle Inequality, [Makarychev et al., 2019])**.** *For any $p \geq 1$ and $\xi \in (0, 1]$, there exists a constant $\vartheta_{p,\xi} > 0$ that depends only on p and ξ such that for any a, b, c in a metric space,*

$$\operatorname{dist}(a, b)^p \leq (1 + \xi) \cdot \operatorname{dist}(a, c)^p + \vartheta_{p,\xi} \cdot \operatorname{dist}(c, b)^p.$$

A.1 Properties of Semi-Coreset

A.1.1 Monge's Optimal Transport and Semi-Coreset

We use a generalization of the optimal transport [Monge, 1781], as defined in [Chang et al., 2021].

Definition 13 (Generalized Transport Cost, [Chang et al., 2021])**.** *Let X and $\tilde{X}$ be weighted point sets over a metric space U. For a mapping $\Psi : U \to U$, the* generalized p-transport cost *is defined as:*

$$\operatorname{mt}(\Psi, X, \tilde{X}) := \sum_{x \in U} w_X(x) \cdot \operatorname{dist}(x, \Psi(x))^p + \sum_{y \in U} |w_X(\Psi^{-1}(y)) - w_{\tilde{X}}(y)|, \tag{5}$$

where we use the notation $w_X(x)$ to denote the weight of point x in X and $w_X(S) = \sum_{x \in S} w_X(x)$.

In other words, instead of allowing just the distance error (first term), we also allow weight mismatch (second term). This make it convenient for DP, which naturally produces noisy weights.

In Chang et al. [2021], it was shown that if two sets have "small" optimal transport cost, then one is a good coreset of the other. However, the "small" cost requirement in Chang et al. [2021] is restrictive: It requires $\operatorname{mt}(\Psi, X, \tilde{X}) \leq \alpha \cdot \operatorname{opt}_X^{k,p} + \beta$ for $\alpha < 1$. On the other hand, our proof will use $\alpha > 1$. Nevertheless, we show that the statement is still true for $\alpha > 1$, albeit with semi-coreset instead of coreset. This is formalized in the lemma below, which will be handy in our proofs.

Lemma 14 (Transport $\implies$ Semi-Coreset)**.** *Assume that all points in U have a pairwise distance bounded by 1 (i.e., diameter ≤ 1). Let X and $\tilde{X}$ be weighted point sets over U. If there exists a mapping $\Psi : X \to U$ such that $\operatorname{mt}(\Psi, X, \tilde{X}) \leq \alpha \cdot \operatorname{opt}_X^{k,p} + \beta$, then $\tilde{X}$ is a (κ, η_1, η_2)-semi-coreset for X with parameters $\kappa = O(1 + \alpha)$, $\eta_1 = O(\alpha)$, and $\eta_2 = O(\beta)$.*

Proof. Consider any set $C \in \binom{U}{k}$ of k centers. For any point $y \in U$, since the diameter of the metric space is bounded by 1, we have $\operatorname{dist}(y, C)^p \leq 1$. By applying the approximate triangle inequality (Lemma 12) with $\xi = 1$, we can bound the cost of C on the estimated set $\tilde{X}$:

$$\begin{aligned}
&\operatorname{cost}_{\tilde{X}}^p(C) \\
&= \sum_{y \in U} w_{\tilde{X}}(y) \cdot \operatorname{dist}(y, C)^p \\
&\leq \sum_{y \in U} w_X(\Psi^{-1}(y)) \cdot \operatorname{dist}(y, C)^p + \sum_{y \in U} |w_{\tilde{X}}(y) - w_X(\Psi^{-1}(y))| \cdot \operatorname{dist}(y, C)^p \\
&\leq \sum_{x \in U} w_X(x) \cdot \operatorname{dist}(\Psi(x), C)^p + \sum_{y \in U} |w_{\tilde{X}}(y) - w_X(\Psi^{-1}(y))| \\
&\leq \sum_{x \in U} w_X(x) \cdot (2 \cdot \operatorname{dist}(x, C)^p + \vartheta_{p,1} \cdot \operatorname{dist}(x, \Psi(x))^p) + \sum_{y \in U} |w_{\tilde{X}}(y) - w_X(\Psi^{-1}(y))| \\
&\leq 2\operatorname{cost}_X^p(C) + \max(\vartheta_{p,1}, 1) \cdot \operatorname{mt}(\Psi, X, \tilde{X}).
\end{aligned}$$

Using the bounds $\mathrm{mt}(\Psi, X, \tilde{X}) \leq \alpha \cdot \mathrm{opt}_X^{k,p} + \beta$ and $\mathrm{opt}_X^{k,p} \leq \mathrm{cost}_X^p(C)$, we obtain the upper bound:

$$\begin{aligned}\mathrm{cost}_{\tilde{X}}^p(C) &\leq 2\mathrm{cost}_X^p(C) + \max(\vartheta_{p,1}, 1)\left(\alpha \cdot \mathrm{opt}_X^{k,p} + \beta\right) \\ &\leq (2 + \max(\vartheta_{p,1}, 1) \cdot \alpha)\,\mathrm{cost}_X^p(C) + \max(\vartheta_{p,1}, 1)\beta.\end{aligned}$$

A symmetric argument upper bounds the true cost:

$$\mathrm{cost}_X^p(C) \leq 2\mathrm{cost}_{\tilde{X}}^p(C) + \max(\vartheta_{p,1}, 1)\left(\alpha \cdot \mathrm{opt}_X^{k,p} + \beta\right).$$

Rearranging this to isolate $\mathrm{cost}_{\tilde{X}}^p(C)$ yields the lower bound:

$$\mathrm{cost}_{\tilde{X}}^p(C) \geq \frac{1}{2}\mathrm{cost}_X^p(C) - \frac{\max(\vartheta_{p,1}, 1) \cdot \alpha}{2}\mathrm{opt}_X^{k,p} - \frac{\max(\vartheta_{p,1}, 1)}{2} \cdot \beta.$$

By defining $\kappa = \max(2 + \max(\vartheta_{p,1}, 1) \cdot \alpha, 2) = O(1 + \alpha)$, $\eta_1 = \frac{\max(\vartheta_{p,1}, 1) \cdot \alpha}{2} = O(\alpha)$, and $\eta_2 = \max(\vartheta_{p,1}, 1) \cdot \beta = O(\beta)$, both the upper and lower bounds of the semi-coreset definition are satisfied. □

A.1.2 Proof of Lemma 5

Proof of Lemma 5. Let C_X^* denote the optimal set of k centers for X such that $\mathrm{cost}_X^p(C_X^*) = \mathrm{opt}_X^{k,p}$. By the definition of a (κ, η_1, η_2)-semi-coreset applied to C_X^*, we have the upper bound

$$\mathrm{cost}_{\tilde{X}}^p(C_X^*) \leq \kappa \cdot \mathrm{cost}_X^p(C_X^*) + \eta_2 = \kappa \cdot \mathrm{opt}_X^{k,p} + \eta_2.$$

Since $\mathrm{opt}_{\tilde{X}}^{k,p} \leq \mathrm{cost}_{\tilde{X}}^p(C)$ for any $C \in \binom{U}{k}$, it follows that $\mathrm{opt}_{\tilde{X}}^{k,p} \leq \mathrm{cost}_{\tilde{X}}^p(C_X^*)$. By assumption, the set C of centers is an (α, β)-approximation for $\tilde{X}$. Therefore,

$$\begin{aligned}\mathrm{cost}_{\tilde{X}}^p(C) &\leq \alpha \cdot \mathrm{opt}_{\tilde{X}}^{k,p} + \beta \\ &\leq \alpha \cdot \mathrm{cost}_{\tilde{X}}^p(C_X^*) + \beta \\ &\leq \alpha \cdot \left(\kappa \cdot \mathrm{opt}_X^{k,p} + \eta_2\right) + \beta \\ &= \alpha\kappa \cdot \mathrm{opt}_X^{k,p} + \alpha\eta_2 + \beta. \end{aligned} \tag{6}$$

Now, we apply the lower bound property of the (κ, η_1, η_2)-semi-coreset to the centers C:

$$\frac{1}{\kappa} \cdot \mathrm{cost}_X^p(C) - \eta_1 \cdot \mathrm{opt}_X^{k,p} - \eta_2 \leq \mathrm{cost}_{\tilde{X}}^p(C). \tag{7}$$

Substituting the upper bound from (6) into (7), we obtain:

$$\frac{1}{\kappa} \cdot \mathrm{cost}_X^p(C) - \eta_1 \cdot \mathrm{opt}_X^{k,p} - \eta_2 \leq \alpha\kappa \cdot \mathrm{opt}_X^{k,p} + \alpha\eta_2 + \beta.$$

Rearranging to isolate $\mathrm{cost}_X^p(C)$ yields:

$$\begin{aligned}\frac{1}{\kappa} \cdot \mathrm{cost}_X^p(C) &\leq (\alpha\kappa + \eta_1) \cdot \mathrm{opt}_X^{k,p} + (\alpha + 1)\eta_2 + \beta \\ \mathrm{cost}_X^p(C) &\leq (\alpha\kappa^2 + \eta_1\kappa) \cdot \mathrm{opt}_X^{k,p} + \kappa(\alpha + 1)\eta_2 + \kappa\beta.\end{aligned}$$

This completes the proof. □

A.2 Consistent k-Clustering via Grid Snapping: Proof of Theorem 6

As mentioned in §2.2, the original consistency bound in Fichtenberger et al. [2021] depends on the *aspect ratio* of the metric space, which is defined as the ratio between the largest distance and the smallest non-zero distance. Their result can be stated as follows:

Theorem 15 (Fichtenberger et al. [2021]). *Let $\mathcal{M} = (U, d)$ be any metric space with aspect ratio Δ. There is an online (k, p)-clustering $(k \cdot \text{polylog}\,(n\Delta))$-consistent algorithm for metric space $\mathcal{M}$ whose output is an $(O(1), 0)$-approximation with high probability. Moreover, the algorithm has time and space complexity at most polynomial in the number of distinct points in the input stream and the time needed to compute distances between points.*

To bypass the aspect ratio assumption in consistent clustering, we transform the input stream by snapping points to a fine grid. We show that this transformation controls the aspect ratio of the active metric space while incurring only a constant additive error, which we can formalize via the generalized transport cost.

Definition 16 (Grid-Snapped Stream). *Let $X = (x_1, \ldots, x_n)$ be a stream of points in $\mathbb{R}^d$ with a metric diameter at most 1. Define the grid $G = (\frac{1}{nd}\mathbb{Z})^d$. Let the snapped stream be $\hat{X} = (\hat{x}_1, \ldots, \hat{x}_n)$, where each $\hat{x}_t$ is obtained by rounding each coordinate of x_t to a multiple of $\frac{1}{nd}$ in such a way that its absolute value does not increase.*

Lemma 17 (Snapped Stream as a Semi-Coreset). *For any $t \in [n]$, the snapped multiset $\hat{X}_{\le t}$ is a $(\kappa, 0, \eta_2)$-semi-coreset for $X_{\le t}$ for the (k, p)-clustering objective, where $\kappa = O(1)$ and $\eta_2 = O(1)$.*

Proof. Consider the natural mapping $\Psi : X_{\le t} \to \hat{X}_{\le t}$ given by $\Psi(x_i) = \hat{x}_i$. Because the point weights match perfectly, the weight difference term in the generalized transport cost (5) is 0.

For the distance term in (5), rounding each coordinate to displaces a point by at most $\frac{1}{nd}$ along any dimension. The ℓ_2 displacement is therefore bounded by:

$$\text{dist}(x_i, \hat{x}_i) \le \sqrt{d\left(\frac{1}{nd}\right)^2} = \frac{\sqrt{d}}{nd} \le \frac{1}{n\sqrt{d}} \le \frac{1}{n}.$$

For the (k, p)-clustering objective, the generalized p-transport cost is:

$$\text{mt}(\Psi, X_{\le t}, \hat{X}_{\le t}) = \sum_{i=1}^{t} \text{dist}(x_i, \hat{x}_i)^p \le t \cdot \frac{1}{n} \le 1.$$

By applying Lemma 14, $\hat{X}_{\le t}$ is an $(O(1), 0, O(1))$-semi-coreset for $X_{\le t}$. □

We are now ready to prove Theorem 6.

Proof of Theorem 6. Instead of clustering the original stream X, we maintain and cluster the grid-snapped stream $\hat{X}$ as defined above (Definition 16). Since all points in $\hat{X}$ are restricted to the grid $G = (\frac{1}{nd}\mathbb{Z})^d$ and the overall diameter is at most 1, the maximum distance between any two points in $\hat{X}$ is 1. Conversely, the minimum non-zero distance between any two distinct grid points is at least $\frac{1}{nd}$. Therefore, the aspect ratio Δ of the active points in $\hat{X}$ is strictly bounded by nd.

We run the consistent k-clustering algorithm from Theorem 15 over the snapped stream $\hat{X}$. Because $\Delta \le nd$, this guarantees a total consistency cost of $k \cdot \text{polylog}\,(n\Delta) \le k \cdot \text{polylog}\,(nd)$.

For the utility guarantee, Theorem 15 ensures that with high probability, for all $t \in [n]$, C_t is an $(O(1), 0)$-approximation on the snapped points $\hat{X}_{\le t}$. By Lemma 17, $\hat{X}_{\le t}$ is a $(O(1), 0, O(1))$-semi-coreset for $X_{\le t}$. Applying Lemma 5, we can thus conclude that C_t is an $(O(1), O(1))$-approximation for the original set $X_{\le t}$, which concludes our proof. □

A.3 Differential Privacy

As mentioned in §2.4, our privacy proof holds in the stronger *adaptive* model as well. Here we follow terminologies from Jain et al. [2023], although similar models were considered even before (e.g. Thakurta and Smith [2013]). The adaptive setting can be defined using a game between the adversary and the algorithm as follows.

Definition 18 (Adaptive continual release game). *Let $\mathcal{B}$ be an adversary and $b \in \{0, 1\}$. The game $\mathsf{Game}^b(\mathcal{B}, \mathcal{M})$ proceeds as follows.*

1. *For $t = 1, \ldots, n$, given the transcript $\pi_{t-1} = (o_1, \ldots, o_{t-1})$, the adversary declares $\text{type}_t \in \{\textsf{regular}, \textsf{challenge}\}$, where* challenge *is declared at most once over the whole game.*
   - *If $\text{type}_t =$* regular*, $\mathcal{B}$ outputs a single x_t, which is sent to $\mathcal{M}$ in both worlds.*
   - *If $\text{type}_t =$* challenge*, set $i^* \leftarrow t$; $\mathcal{B}$ outputs a pair (x_t^0, x_t^1) with $\perp \in \{x_t^0, x_t^1\}$, and x_t^b is sent to $\mathcal{M}$.*
2. *$\mathcal{M}$ releases o_t, which is appended to the transcript.*
3. *The game outputs π_n.*

Definition 19 (DP under adaptive inputs)**.** *$\mathcal{M}$ is (ε, δ)-DP in the adaptive continual release model if for every adversary $\mathcal{B}$ and every measurable set T of transcripts,*

$$\Pr\left[\mathsf{Game}^0(\mathcal{B}, \mathcal{M}) \in T\right] \;\le\; e^{\varepsilon} \Pr\left[\mathsf{Game}^1(\mathcal{B}, \mathcal{M}) \in T\right] + \delta.$$

It is clear that Definition 19 is no weaker than that in Definition 8: For a fixed pair neighboring streams $X = (x_1, \ldots, x_n), X' = (x'_1, \ldots, x'_n)$, the adaptive adversary can always simply set $x_t^0 = x_t$ and $x_t^1 = x'_t$ for all $t \in [n]$ and recovers the non-adaptive guarantee.

Henceforth, for online algorithms, we will always consider privacy in the stronger adaptive continual release model (Definition 19). For brevity, we will sometimes not state this explicitly.

A.3.1 Tools from Differential Privacy

We next state two lemmas which will be helpful in the privacy proofs. First is the so-called "post-processing" property of DP, specialized to the adaptive continual release model below.

Lemma 20 (Online post-processing)**.** *Let $\mathcal{M}$ be (ε, δ)-DP in the adaptive continual release model, and let $\mathcal{M}'$ be the online mechanism that runs $\mathcal{M}$ internally and, at step t, releases $g_t(\pi_t; \sigma)$, where $\pi_t = (o_1, \ldots, o_t)$ is $\mathcal{M}$'s transcript prefix, g_t is a fixed function, and σ is randomness independent of the input. Then $\mathcal{M}'$ is (ε, δ)-DP in the adaptive continual release model.*

We also need composition properties of DP. Since we employ multiple online DP subroutines which are interleaved, we require a "concurrent" version of DP composition, as stated below.

Lemma 21 (Concurrent composition; Henzinger et al. [2026])**.** *Let $\mathcal{M}_1, \ldots, \mathcal{M}_m$ be continual mechanisms run concurrently, where each $\mathcal{M}_i$ is $(\varepsilon_i, \delta_i)$-DP in the adaptive model, with respect to a neighbor relation $\sim_i$ on its input streams. Consider an adversary that interacts with all m mechanisms concurrently, interleaving its interactions arbitrarily and choosing each input to each mechanism adaptively as a function of all outputs received so far from all mechanisms. Then the composition is $\left(\sum_i \varepsilon_i,\ \sum_i \delta_i\right)$-DP with respect to any pair of adaptive strategies whose induced input streams satisfy $\sim_i$ for every i.*

We also need a "parallel" composition theorem for continual mechanisms, in which the neighbor relation on the composed input is restricted so that the two worlds differ only in the inputs to a single mechanism. Under this restriction, the privacy loss does not grow with m. We state this for the pure-DP case, which is all we need.

Lemma 22 (Parallel composition; Henzinger et al. [2026])**.** *Consider the setting of Lemma 21, with the neighbor relation restricted as follows: the input streams induced by the two strategies are identical for all but one mechanism $\mathcal{M}_{i^*}$, and neighboring for $\mathcal{M}_{i^*}$. If $\varepsilon_1 = \cdots = \varepsilon_m = \varepsilon'$ and $\delta_1 = \cdots = \delta_m = 0$, then the composed mechanism is ε'-DP with respect to this relation.*

A.3.2 Differentially Private Continual Counting

For continual counting, two input streams $Y = (y_1, \ldots, y_n) \in \{0,1\}^n$ and $Y' = (y'_1, \ldots, y'_n) \in \{0,1\}^n$ are neighbors iff they differ on a single coordinate.

Throughout, we will assume that the estimates e_t of $\sum_{i \in [t]} y_i$ are monotone, i.e., $e_1 \le \cdots \le e_n$, and that they are integers. (If this does not hold, it can be easily achieved by taking $\lceil \max_{i \in [t]} e_i \rceil$ instead of e_t.) The seminal work of Dwork et al. [2010], Chan et al. [2011] gives a DP algorithm that is $O(\text{polylog}\,(n))$-accurate with high probability. This is stated more formally below.

Theorem 23 (Dwork et al. [2010], Chan et al. [2011])**.** *There exists an ε-DP[10] continual counting algorithm that, for any $\gamma > 0$, is $O\left(\frac{1}{\varepsilon} \cdot \log^{1.5} n \cdot \log(1/\gamma)\right)$-accurate with probability $1 - \gamma$. The total running time of the algorithm is $O(n \log n)$ and its space complexity is $O(\log n)$.*

A.4 Dense Ball Algorithm

For the dense ball problem (Definition 7), we consider two input datasets S, S' neighbors iff $|S \Delta S'| \leq 1$, where Δ denote the symmetric difference between the two sets.

Our definition of the dense ball problem is closely related to the 1-Cluster and the densest ball problems previously studied in DP literature [Nissim et al., 2016, Nissim and Stemmer, 2018, Ghazi et al., 2020]. By repeatedly applying a known algorithm for this problem, it is simple to derive the following result for the dense ball problem.

Theorem 24. *Let $N \in \mathbb{N}$ and suppose that the input S has size at most N. Then, there exists an (ε, δ)-DP algorithm with time and space complexity $(Nd \cdot \log(1/r_{\min}))^{O(1)}$ and size blow-up of $O(1)$, which solves the $(r_{\min}, \tau)$-dense ball with probability $1 - \gamma$ where*

- $\tau = O\left(\left(\frac{d \cdot N}{\varepsilon}\right)^{1/2} \cdot \text{polylog}\left(\frac{Nd}{\varepsilon r_{\min}}\right)\right)$ *for $\delta = 0$, and,*
- $\tau = O\left(\left(\frac{d \cdot N}{\varepsilon^2}\right)^{1/3} \cdot \text{polylog}\left(\frac{Nd}{\varepsilon \delta r_{\min}}\right)\right)$ *for $\delta > 0$.*

Our construction builds upon the DP 1-Cluster algorithms of Ghazi et al. [2020]. We formally define the problem and recall their result below.

Definition 25 (1-Cluster, [Nissim et al., 2016])**.** *Given a multiset S of size at most N in a metric space, a target number of points τ, a minimum radius, and an approximation ratio $\alpha \geq 1$, the* 1-Cluster problem *asks to output a center c and a radius r such that the ball $\mathcal{B}(c, r)$ contains at least $\tau - t$ points from S (where t is the additive error), and $r \leq \alpha \cdot \max\{r_{\min}, r_{\text{opt}}\}$, where r_{opt} is the minimum radius of a ball containing at least τ points in S.*

Theorem 26 (DP 1-Cluster, [Ghazi et al., 2020])**.** *There exist (ε, δ)-DP algorithms for the 1-Cluster problem in the Euclidean space with approximation ratio $\alpha \in (1, 2)$ that succeed with probability $1 - \beta$ and have an additive error bounded by:*

- $t = O\left(\frac{d}{\varepsilon} \cdot \text{polylog}\left(\frac{Nd}{\varepsilon r_{\min} \beta}\right)\right)$ *for pure-DP ($\delta = 0$).*
- $t = O\left(\frac{\sqrt{d}}{\varepsilon} \cdot \text{polylog}\left(\frac{Nd}{\varepsilon \delta r_{\min} \beta}\right)\right)$ *for approximate-DP ($\delta > 0$).*

The time and space complexities are bounded by $(Nd \cdot \log(1/r_{\min}))^{O(1)}$.

We quickly note that, in the case of approximate-DP, the algorithm from [Ghazi et al., 2020] in fact has an even better dependency on $r_{\min}$. However, we choose to state this coarser error bound since it does not effect our final guarantee anyway as we only choose $r_{\min} = \frac{1}{n}$ in the subsequent step.

We formalize our procedure in Algorithm 3. The algorithm iteratively extracts dense balls from the dataset by repeatedly invoking the DP 1-Cluster subroutine.

Algorithm 3 Dense Ball via Iterative 1-Cluster

Require: Multiset S of size at most N, minimum target radius $r_{\min}$, target points τ, privacy parameters (ε, δ), failure probability γ.

$O \leftarrow \emptyset$
$S_1 \leftarrow S$
$K \leftarrow \lceil 2N/\tau \rceil$ {Number of iterations}
Set per-run privacy parameters (ε', δ') based on composition over K total invocations.
for $i = 1, \ldots, K$ **do**
  $(c_i, r_i) \leftarrow$ Result of running the 1-Cluster algorithm of (Theorem 26) on S_i target τ, approximation ratio $\alpha = O(1)$, failure probability $\beta = \gamma/K$, and privacy (ε', δ').
  $O \leftarrow O \cup \{c_i\}$
  $S_{i+1} \leftarrow S_i \setminus \mathcal{B}(c_i, r_i)$
return O

[10] Here the DP definition as in Definition 19 with 0 being treated as $\perp$.

Proof of Theorem 24. We analyze Algorithm 3 by establishing its privacy, accuracy (size bound and coverage), and matching the required parameter bounds for τ.

Privacy. Algorithm 3 invokes the base 1-Cluster algorithm a total of K times. We distribute the total privacy budget (ε, δ) across all $K = O(N/\tau)$ invocations using composition theorems:

- Pure-DP: By basic composition, setting $\varepsilon' = \varepsilon/K$ ensures that Algorithm 3 is ε-DP.
- Approximate-DP: By advanced composition, to achieve overall (ε, δ)-DP, it is sufficient to set $\delta' = \Theta\left(\frac{\delta}{K}\right)$ and $\varepsilon' = \Theta\left(\frac{\varepsilon}{\sqrt{K \ln(1/\delta)}}\right)$.

Size Bound. The algorithm runs for exactly K steps, outputting exactly one center per step. Thus, the output set O has size $|O| = K \leq 4N/\tau$. This satisfies the size bound property with blow-up factor $\Upsilon = 4$.

Coverage. Let t denote the additive error of the 1-Cluster algorithm. As specified in more detail below, we will parameterize the noise such that $t \leq \tau/2$.

Since each run of the 1-Cluster algorithm is successful with probability $1 - \beta$, all the runs succeed with probability at least $1 - \beta \cdot K = 1 - \gamma$. Conditioned on this, we will show that the coverage constraint is satisfied. Suppose there exist $u \in \mathbb{R}^d, r \geq r_{\min}$ such that $|S \cap \mathcal{B}(u, r)| \geq \tau$. We must show that $\operatorname{dist}(u, O) \leq F \cdot r$.

First, let us assume for contradiction that no removed ball $\mathcal{B}(c_i, r_i)$ intersects $\mathcal{B}(u, r)$. If this were true, none of the points in $S \cap \mathcal{B}(u, r)$ would ever be removed. Thus, at every step i, the remaining set S_i would contain all $\geq \tau$ points of $S \cap \mathcal{B}(u, r)$. Because the algorithm succeeds at every step and $t \leq \tau/2$, each of the K steps removes at least $\tau/2$ points from the dataset. Across K steps, the algorithm would remove $K \cdot (\tau/2) = \lceil 2N/\tau \rceil \cdot (\tau/2) \geq N$ points. Since the total dataset has size at most N, this implies all points are eventually removed, contradicting the assumption that the points in $\mathcal{B}(u, r)$ remain untouched.

Therefore, there must be at least one step where the removed ball $\mathcal{B}(c_i, r_i)$ intersects $\mathcal{B}(u, r)$. Let j be the *first* such step. At the start of step j, none of the points in $\mathcal{B}(u, r)$ have been removed, meaning $|S_j \cap \mathcal{B}(u, r)| \geq \tau$. This implies that the optimal radius r_{opt} for capturing τ points in S_j is bounded by r (i.e., $r_{\text{opt}} \leq r$). Since we ran 1-Cluster with approximation ratio $\alpha = O(1)$, the output radius guarantees $r_j \leq \alpha \cdot \max\{r_{\min}, r_{\text{opt}}\} \leq \alpha \cdot r$. Since $\mathcal{B}(c_j, r_j)$ intersects $\mathcal{B}(u, r)$, let x be a point in their intersection. By the triangle inequality, we have:

$$\operatorname{dist}(u, c_j) \leq \operatorname{dist}(u, x) + \operatorname{dist}(x, c_j) \leq r + r_j \leq 3r.$$

Since $c_j \in O$, this proves the required coverage.

Parameter Settings. To complete the proof, we select the parameters to ensure $t \leq \tau/2$.

For Pure-DP: By Theorem 26, the additive error of 1-Cluster is $t = O\left(\frac{d}{\varepsilon'} \cdot \operatorname{polylog}\left(\frac{Nd}{\varepsilon' r_{\min} \gamma}\right)\right)$. Substituting $\varepsilon' = \varepsilon/K$, we get:

$$t = O\left(\frac{dN}{\varepsilon\tau} \cdot \operatorname{polylog}\left(\frac{Nd}{\varepsilon r_{\min} \gamma}\right)\right)$$

Enforcing $t \leq \tau/2$ yields the requirement:

$$\tau^2 \geq \Theta\left(\frac{dN}{\varepsilon} \cdot \operatorname{polylog}\left(\frac{Nd}{\varepsilon r_{\min} \gamma}\right)\right) \iff \tau \geq \Theta\left(\left(\frac{dN}{\varepsilon}\right)^{1/2} \cdot \operatorname{polylog}\left(\frac{Nd}{\varepsilon r_{\min} \gamma}\right)\right).$$

For Approximate-DP: By Theorem 26, the additive error is $t = O\left(\frac{\sqrt{d}}{\varepsilon'} \cdot \operatorname{polylog}\left(\frac{Nd}{\varepsilon\delta r_{\min} \gamma}\right)\right)$. Substituting our advanced composition parameter $\varepsilon' = \Theta\left(\frac{\varepsilon}{\sqrt{K \log(1/\delta)}}\right) = \Theta\left(\frac{\varepsilon\sqrt{\tau}}{\sqrt{N \log(1/\delta)}}\right)$, we get:

$$t = O\left(\frac{\sqrt{dN}}{\varepsilon\sqrt{\tau}} \cdot \operatorname{polylog}\left(\frac{Nd}{\varepsilon\delta r_{\min} \gamma}\right)\right)$$

Enforcing $t \le \tau/2$ yields the requirement:

$$\tau^{3/2} \ge \Theta\left(\frac{\sqrt{dN}}{\varepsilon}\operatorname{polylog}\left(\frac{Nd}{\varepsilon\delta r_{\min}\gamma}\right)\right) \iff \tau \ge \Theta\left(\left(\frac{dN}{\varepsilon^2}\right)^{1/3}\operatorname{polylog}\left(\frac{Nd}{\varepsilon\delta r_{\min}\gamma}\right)\right).$$

Time and Space Complexity. The 1-Cluster subroutine is called $K \le N$ times. Each run takes $(Nd\log(1/r_{\min}))^{O(1)}$ time and space. Thus, the total time and space complexity is bounded by $(Nd\log(1/r_{\min}))^{O(1)}$. □

B Missing Proofs from §3.1

B.1 Private Meyerson's Sketch: Privacy and Efficiency

In this section, we state generic bounds on the privacy and efficiency guarantees of our DP Meyerson's sketch (Algorithm 2).

Lemma 27 (Privacy)**.** *If $\mathcal{A}_{\text{dense}}$ is $(\varepsilon_d, \delta_d)$-DP and $\mathcal{A}_{\text{count}}$ is $(\varepsilon_c, \delta_c)$-DP, then Algorithm 2 is $\left(2\cdot\Psi\cdot\varepsilon_d + \varepsilon_c, 2\cdot e^{2\varepsilon_d}\cdot\Psi\cdot\delta_d + \delta_c\right)$-DP.*

Proof. Fix an adversary $\mathcal{B}$. Recall that we use i^* to denote the challenge index.

First, it will be more convenient to consider a modified algorithm $\widehat{\mathcal{A}}$, which is exactly the same as Algorithm 2, except it releases (e_t, R_t) at each step (instead of just R_t). By Lemma 20, it suffice to prove that $\widehat{\mathcal{A}}$ is $\left(2\cdot\Psi\cdot\varepsilon_d + \varepsilon_c, 2\cdot e^{2\varepsilon_d}\cdot\Psi\cdot\delta_d + \delta_c\right)$-DP.

Moreover, in Algorithm 2, we write $Z_t \sim \text{Ber}(\lambda_t)$. For our purpose, it will be more transparent to instead think of this as setting $Z_t = \mathbf{1}[U_t \le \lambda_t]$ where $U = (U_1, \ldots, U_n)$ is drawn i.i.d. uniform on $[0,1]$. Let $\widehat{\mathcal{A}}_u$ denote $\widehat{\mathcal{A}}$ with U fixed to u. By the convexity of DP (see, e.g., Desfontaines and Pejó [2020]), it suffices to prove the bound for every fixed u.

Fix $u = (u_1, \ldots, u_n)$ and a transcript prefix $\pi_{t-1} = ((e_1, R_1), \ldots, (e_{t-1}, R_{t-1}))$. Given π_{t-1}, the set $R_{\le t-1}$ and the counter j are determined, hence so is the map $\lambda(\cdot) = \min\{1, \frac{\tau k L}{\text{opt}_{\text{est}}}\text{dist}(\cdot, R_{\le t-1})^p\}$. The adversary is also a function of π_{t-1}, so it submits the same x_t in both worlds for $t \ne i^*$. Therefore, the bit fed to $\mathcal{A}_{\text{count}}$ at step t is the deterministic function $z_t = \mathbf{1}[u_t \le \lambda(x_t)]$ of the submitted point, and we have the following. (We use the superscript $b \in \{0,1\}$ to denote the execution on $(x_1^b, \ldots, x_n^b)$.)

(a) *The counting streams are neighbors.* For $t \ne i^*$, we have $x_t^0 = x_t^1$ and hence $z_t^0 = z_t^1$. So the two streams differ in at most the coordinate i^*.

(b) *The dense-ball inputs are 2-neighbors.* Assume w.l.o.g. that $x_{i^*}^1 = \perp$. Both executions insert points in stream order, skipping an arrival once $|S| = \Psi\tau$. We thus have $S_t^0 \setminus S_t^1 \subseteq \{x_{i^*}^0\}$ and $|S_t^1 \setminus S_t^0| \le 1$, Hence, $|S_t^0 \,\Delta\, S_t^1| \le 2$.

Note that $\widehat{\mathcal{A}}_u$ is the concurrent composition of $\mathcal{A}_{\text{count}}$, run on a bit stream generated adaptively from the transcript, with at most Ψ invocations of $\mathcal{A}_{\text{dense}}$, each on an input determined by the transcript, u and the submitted points. By (a), the first component is $(\varepsilon_c, \delta_c)$-DP; by (b) and group privacy for groups of size 2, each invocation of $\mathcal{A}_{\text{dense}}$ is $(2\varepsilon_d, 2e^{2\varepsilon_d}\delta_d)$-DP. Lemma 21 then gives the claimed privacy guarantee. □

Lemma 28 (Efficiency)**.** *Suppose that $\mathcal{A}_{\text{count}}$ runs in $\mathcal{T}_c(n)$ time and uses $\mathcal{S}_c(n)$ space, and $\mathcal{A}_{\text{dense}}$ runs in $\mathcal{T}_d(n)$ time and uses $\mathcal{S}_d(n)$ space. Furthermore, suppose that the output stream $O_{\le n}$ has size at most K. Then, Algorithm 2 runs in $O\left(ndK + \mathcal{T}_c(n) + \Psi\cdot\mathcal{T}_d(\Psi\cdot\tau)\right)$ time and uses $O\left(\Psi\cdot(\tau + \Psi\cdot\Upsilon)\cdot d + \mathcal{S}_c(n) + \mathcal{S}_d(\Psi\cdot\tau)\right)$ space.*

Proof. For the running time, observe that all calculations outside the calls to $\mathcal{A}_{\text{count}}, \mathcal{A}_{\text{dense}}$ takes $O(d)$ time per point except for the computation of the closest point (Line 5) which may take up to $O(K\cdot d)$ time. Furthermore, we only invoke $\mathcal{A}_{\text{dense}}$ at most $O(\Psi)$ times, and each time the input S_t has size at most $\Psi\cdot\tau$. The total running time thus follows.

As for the space complexity, outside of $\mathcal{A}_{\text{count}}, \mathcal{A}_{\text{dense}}$, the only thing needed to be kept by our algorithm are S_t, R_t. The former has size at most $\Psi \cdot \tau$ and the latter has size at most $\Psi^2 \cdot \Upsilon$ (due to the size blow-up guarantee of $\mathcal{A}_{\text{dense}}$). Finally, note that each element in S_t, R_t are points in $\mathbb{R}^d$ so it requires $O(d)$ space. □

B.2 Combined Prefix-Approximation Guarantee via Guessing

By using a randomized guessing strategy for the estimated optimum cost, we can eliminate the dependence on opt_{est} in the error bounds of the Private Meyerson's Sketch. This allows us to strictly parameterize the capacity bound Ψ and the density threshold τ in terms of k, d, n.

Algorithm 4 Random-Guess DP Meyerson's Sketch $\mathcal{A}_{\text{guess-bicri}}$

Require: Stream $X = (x_1, \ldots, x_n)$ of points in a metric space $\mathcal{M} = (U, d)$ with diameter ≤ 1.

1: **Parameters:** Target centers k, Privacy budgets (ε, δ).
2: Define the set of possible guesses: $\mathcal{G} = \{2^{\lceil \log_2 n \rceil - 1}, 2^{\lceil \log_2 n \rceil - 2}, \ldots, 2^{-\lceil \log_2 n \rceil}\}$
3: Sample an estimated optimum opt_{est} uniformly at random from $\mathcal{G}$.
4: Set the target sketch capacity: $\Psi = \Theta(k \log n)$
5: Set the density threshold τ depending on the privacy model:
- For pure-DP: $\tau = \Theta\left(\frac{d \cdot k^2}{\varepsilon} \text{polylog}\left(\frac{nd}{\varepsilon}\right)\right)$
- For approximate-DP: $\tau = \Theta\left(\frac{\sqrt{d} \cdot k^{1.5}}{\varepsilon} \text{polylog}\left(\frac{nd}{\varepsilon\delta}\right)\right)$

6: Instantiate $\mathcal{A}_{\text{count}}$ as the $(\varepsilon/2, \delta/2)$-DP continual counting algorithm.
7: Instantiate $\mathcal{A}_{\text{dense}}$ as the $(\frac{\varepsilon}{4\Psi}, \frac{\delta}{8\Psi e^{\varepsilon/(2\Psi)}})$-DP dense ball algorithm.
8: Run $\mathcal{A}_{\text{Meyerson}}(\text{opt}_{\text{est}}, \Psi)$ on X using the configured parameters.
9: **return** Output stream $(O_1, \ldots, O_n)$ from the sketch.

Lemma 29 (DP Prefix-Approximation)**.** *Given a stream $X = (x_1, \ldots, x_n)$ of n points and a target number of centers k, Algorithm 4 is (ε, δ)-DP. Furthermore, it outputs a multiset-stream $(O_1, \ldots, O_n)$ satisfying:*

1. ***Accuracy:*** *For a fixed $t \in [n]$, with probability at least $\Omega\left(\frac{1}{\log n}\right)$, $(O_1, \ldots, O_t)$ is an $(O(1), \beta)$-prefix-approximation of $(x_1, \ldots, x_t)$, where the additive error β is bounded by:*
   - *$\beta = O\left(\frac{d \cdot k^3}{\varepsilon} \text{polylog}\left(\frac{nd}{\varepsilon}\right)\right)$ for pure-DP.*
   - *$\beta = O\left(\frac{\sqrt{d} \cdot k^{2.5}}{\varepsilon} \text{polylog}\left(\frac{nd}{\varepsilon\delta}\right)\right)$ for approximate-DP.*

   [nosep]
2. ***Size Bound:*** *The total number of output centers is bounded by $|O_{\leq n}| \leq O(k^2 \log^2 n)$.*

The algorithm processes the stream in total time $\tilde{O}\left(ndk^2\right) + \Lambda^{O(1)}$ and uses space $\Lambda^{O(1)}$, where

- *$\Lambda = \frac{kd \log n}{\varepsilon}$ for pure-DP.*
- *$\Lambda = \frac{kd \log(n/\delta)}{\varepsilon}$ for approximate-DP.*

Proof. The privacy guarantee of Algorithm 4 immediately follows from that of $\mathcal{A}_{\text{Meyerson}}$ (Lemma 27).

Accuracy Guarantee (Conditioning on a Correct Guess). Fix $t \in [n]$. Let $\text{opt} := \text{opt}^{k,p}_{X_{\leq t}}$ denote the true optimal cost. Since the metric space has diameter ≤ 1 and there are $t \leq n$ points, $\text{opt} \leq n$. Let us define the "correct guess" $g^* \in \mathcal{G}$ as follows:

- If $\text{opt} > 1/n$, let g^* be the unique power of 2 in $\mathcal{G}$ such that $g^* < \text{opt} \leq 2g^*$.
- If $\text{opt} \leq 1/n$, let $g^* = 2^{-\lceil \log_2 n \rceil} \leq 1/n$.

Since $\mathcal{G}$ contains exactly $2\lceil \log_2 n \rceil$ elements, the algorithm selects $\text{opt}_{\text{est}} = g^*$ with probability exactly $\frac{1}{2\lceil \log_2 n \rceil} = \Omega\left(\frac{1}{\log n}\right)$. Conditioned on the event that $\text{opt}_{\text{est}} = g^*$, we now verify that the parameters assigned strictly satisfy the prerequisites for the prefix-approximation guarantee of the private Meyerson's sketch in Lemma 10:

1. **Capacity Ψ:** The prerequisite is $\Psi \geq D \cdot kL\left(1+\frac{\text{opt}}{\text{opt}_{\text{est}}}\right)$ where $L = \Theta\left(\log_2(n/\text{opt}_{\text{est}})\right)$. Because g^* is the correct guess, we have $\frac{\text{opt}}{\text{opt}_{\text{est}}} \leq \frac{\text{opt}}{g^*} \leq 2$ (or $\frac{\text{opt}}{1/n} \leq 1$). Since $\text{opt}_{\text{est}} \geq 1/n$, $L \leq \Theta(\log n)$. Thus, it suffices to choose $\Psi = \Theta(k \log n)$ where $\Theta(\cdot)$ hides a sufficiently large constant.
2. **Threshold τ:** We require both $\mathcal{A}_{\text{count}}$ and $\mathcal{A}_{\text{dense}}$ to succeed with failure probability $\gamma \leq 1/(4n)$.
   - $\mathcal{A}_{\text{count}}$ requires an additive error of $\leq \tau/2$. Given $\varepsilon_c = \varepsilon/2$, by Theorem 23, it suffices to satisfy $\tau \geq \Omega\left(\frac{1}{\varepsilon} \cdot \text{polylog}\left(n\right)\right)$.
   - $\mathcal{A}_{\text{dense}}$ processes datasets of size up to $N = \Psi \cdot \tau$.
     - **For pure-DP:** By Theorem 24, the subroutine requires $\tau \geq \Theta\left(\left(\frac{d\cdot(\Psi\tau)}{\varepsilon_d}\right)^{1/2} \cdot \text{polylog}\left(\frac{nd}{\varepsilon_d}\right)\right)$. Since $\varepsilon_d = \varepsilon/(4\Psi)$, this resolves to:

$$\tau^2 \geq \Theta\left(\frac{d\Psi\tau}{\varepsilon/\Psi} \cdot \text{polylog}\left(\frac{nd}{\varepsilon}\right)\right) \iff \tau \geq \Omega\left(\frac{d\Psi^2}{\varepsilon} \cdot \text{polylog}\left(\frac{nd}{\varepsilon}\right)\right)$$

     Substituting $\Psi = \Theta(k \log n)$, it suffices to take $\tau = \Theta\left(\frac{dk^2}{\varepsilon} \cdot \text{polylog}\left(\frac{nd}{\varepsilon}\right)\right)$.
     - **For approximate-DP:** By Theorem 24, this requires $\tau \geq \Theta\left(\left(\frac{d\cdot(\Psi\tau)}{\varepsilon_d^2}\right)^{1/3} \cdot \text{polylog}\left(\frac{nd}{\varepsilon_d\delta}\right)\right)$. This resolves to:

$$\tau^3 \geq \Theta\left(\frac{d\Psi\tau}{(\varepsilon/\Psi)^2} \cdot \text{polylog}\left(\frac{nd}{\varepsilon\delta}\right)\right) \iff \tau^2 \geq \Theta\left(\frac{d\Psi^3}{\varepsilon^2} \cdot \text{polylog}\left(\frac{nd}{\varepsilon\delta}\right)\right)$$
$$\iff \tau \geq \Theta\left(\frac{\sqrt{d}\Psi^{1.5}}{\varepsilon} \cdot \text{polylog}\left(\frac{nd}{\varepsilon\delta}\right)\right)$$

     Substituting $\Psi = \Theta(k \log n)$, it suffices to take $\tau = \Theta\left(\frac{\sqrt{d}k^{1.5}}{\varepsilon} \cdot \text{polylog}\left(\frac{nd}{\varepsilon\delta}\right)\right)$.

Since all parameter prerequisites are met, the private Meyerson's sketch guarantees that with probability at least $1/2$, the output $(O_1, \ldots, O_t)$ is an $(O(1), \beta)$-prefix-approximation of $(x_1, \ldots, x_t)$, where the additive error bound is given by:

$$\beta_{bicri} = O(\text{opt}_{\text{est}} + \tau kL).$$

Recall that $\text{opt}_{\text{est}} \leq \max(\text{opt}, 1/n)$; thus, the term $O(\text{opt}_{\text{est}})$ scales as $O(\text{opt}) + O(1/n)$. The $O(\text{opt})$ component acts as a constant multiplier absorbing directly into the $O(1)$-multiplicative approximation factor. The $O(1/n)$ term is strictly subsumed by the dominant term $O(\tau kL)$. Thus, the final additive error is purely $\beta = O(\tau kL) = O(\tau k \log n)$. Substituting our values for τ:

- **Pure-DP:** $\beta = O\left(\frac{dk^2}{\varepsilon} \cdot \text{polylog}\left(\frac{nd}{\varepsilon}\right) \cdot k \log n\right) = O\left(\frac{dk^3}{\varepsilon} \cdot \text{polylog}\left(\frac{nd}{\varepsilon}\right)\right)$.
- **Approximate-DP:** $\beta = O\left(\frac{\sqrt{d}k^{1.5}}{\varepsilon} \cdot \text{polylog}\left(\frac{nd}{\varepsilon\delta}\right) \cdot k \log n\right) = O\left(\frac{\sqrt{d}k^{2.5}}{\varepsilon} \cdot \text{polylog}\left(\frac{nd}{\varepsilon\delta}\right)\right)$.

Conditioned on guessing g^*, the algorithm succeeds with probability $1/2$. Thus, the overall unconditional success probability is strictly $\Omega(1/\log n) \cdot \frac{1}{2} = \Omega\left(\frac{1}{\log n}\right)$.

Size Bound and Complexity. By the constraints of the private Meyerson's sketch, the total number of output centers K is bounded by $\Psi^2 \cdot \Upsilon$. Because $\Upsilon = O(1)$ for the 1-Cluster to dense ball reduction, we have $K \leq O(\Psi^2) = O(k^2 \log^2 n)$.

The time and space complexity follows by plugging the above parameters into Lemma 28 with $\mathcal{T}_c(n) = O(n \log n), \mathcal{S}_c(n) = O(n)$ (from Theorem 23) and $\mathcal{T}_d(N) = \mathcal{S}_c(N) = N^{O(1)}$ (from Theorem 24) with $N \leq \Psi \cdot \tau$. □

B.3 Boosting Probability via Repetition

Since Lemma 29 only yields a low-probability guarantee for a single $t \in [n]$, we need to boost its success probability. Fortunately, this can be easily accomplished by repeatedly running the algorithm multiple times. We start by describing the generic reduction below.

Lemma 30. *Suppose that there is an $(\tilde{\varepsilon}, \tilde{\delta})$-DP algorithm that satisfies the following:*

- *Its output stream size is bounded by B.*
- *Its space and time complexity are $\mathcal{S}(n)$ and $\mathcal{T}(n)$.*
- *For each fixed $t \in [n]$, the output stream $(O_1, \ldots, O_t)$ is an (α, β)-prefix-approximation of the input stream $(x_1, \ldots, x_t)$ with probability at least q.*

Then, for some $R = O\left(\frac{\log(n/\gamma)}{q}\right)$ there is an $\left(R \cdot \tilde{\varepsilon}, R \cdot \tilde{\delta}\right)$-DP algorithm that satisfies the following:

- *Its output stream size is bounded by $R \cdot B$.*
- *Its space and time complexity are $O(R) \cdot \mathcal{S}(n)$ and $O(R) \cdot \mathcal{T}(n)$.*
- *With probability $1 - \gamma$, for all $t \in [n]$, the output stream $(O_1, \ldots, O_t)$ is an (α, β)-prefix-approximation of the input stream $(x_1, \ldots, x_t)$.*

Proof. This follows by simply running the algorithm $R = \left\lceil \frac{\ln(n/\gamma)}{q} \right\rceil$ times, and outputting the union of all the output. The output size bound, and space and time complexity obviously grows by a factor of R. As for the (α, β)-prefix-approximation guarantee, observe that it holds if at least one of the repetition satisfies the (α, β)-prefix-approximation. Thus, by a union bound, the probability that it satisfies (α, β)-prefix-approximation for all $t \in [n]$ is at least $1 - n \cdot (1 - q)^R \geq 1 - \gamma$. □

Combining Lemma 30 with Lemma 29 (where $q = \Omega(1/\log n)$), we immediately get the following:

Theorem 31 (High-Probability DP Prefix-Approximation)**.** *Given a stream $X = (x_1, \ldots, x_n)$ of n points and a target number of centers k, there is an (ε, δ)-DP algorithm whose multiset-stream $(O_1, \ldots, O_n)$ satisfies:*

1. ***Accuracy:** With probability at least $1 - \gamma$, for all $t \in [n]$, the output stream $(O_1, \ldots, O_t)$ is an $(O(1), \beta)$-prefix-approximation of the input stream $(x_1, \ldots, x_t)$, where the additive error β is bounded by:*
   - *$\beta = O\left(\frac{d \cdot k^3}{\varepsilon} \operatorname{polylog}\left(\frac{nd}{\varepsilon\gamma}\right)\right)$ for pure-DP.*
   - *$\beta = O\left(\frac{\sqrt{d} \cdot k^{2.5}}{\varepsilon} \operatorname{polylog}\left(\frac{nd}{\varepsilon\delta\gamma}\right)\right)$ for approximate-DP.*
2. ***Size Bound:** The total number of output centers is bounded by $|O_{\leq n}| \leq O(k^2 \cdot \operatorname{polylog}(n/\gamma))$.*

The algorithm processes the stream in total time $\tilde{O}\left(ndk^2 \cdot \log(1/\gamma)\right) + \Lambda^{O(1)}$ and uses space $\Lambda^{O(1)}$, where

- *$\Lambda = \frac{kd \log(n/\gamma)}{\varepsilon}$ for pure-DP.*
- *$\Lambda = \frac{kd \log\left(\frac{n}{\gamma\delta}\right)}{\varepsilon}$ for approximate-DP.*

C From Prefix-Approximation to DP (Semi-)Coreset: Proof of Theorem 9

In this section, we present our algorithm for constructing semi-coreset and prove our main theorem (Theorem 9). As explained in § 3.1, we simply run the online prefix-approximation algorithm and then snap x_t to the closest centers among those produced by the algorithm. We use DP continual counting algorithm to estimate the weight of each center.

C.1 Properties of the DP Semi-Coreset Algorithm: Privacy and Efficiency

The privacy and efficiency guarantee of Algorithm 5 are fairly simple to derive and are stated more formally below.

Lemma 32 (Privacy)**.** *If $\mathcal{A}_{\text{pref}}$ is $(\varepsilon_b, \delta_b)$-DP and $\mathcal{A}_{\text{count}}$ is ε_c-DP, then Algorithm 5 is $(\varepsilon_b + \varepsilon_c, \delta_b)$-DP.*

Algorithm 5 Private Semi-Coreset $\mathcal{A}_{\text{dpcoreset}}$

Require: Stream $X = (x_1, \ldots, x_n)$
Parameters: Online Prefix-Approximation Algorithm $\mathcal{A}_{\text{pref}}$; Continual Counting Algorithm $\mathcal{A}_{\text{count}}$

for $t = 1, \ldots, n$ **do**
  {Do the following upon receiving x_t}
  Feed x_t to $\mathcal{A}_{\text{pref}}$ and let $y_t \subseteq U$ be the output
  for $u \in y_t \setminus Y_{\leq t-1}$ **do**
    $\mathcal{A}^u_{\text{count}} \leftarrow$ start $\mathcal{A}_{\text{count}}$ instance for u
    $e^u_{t-1} \leftarrow 0$
  $\tilde{x}_t \leftarrow \emptyset$ {Output in this time step}
  if $x_t \neq \perp$ **then**
    $f_t \leftarrow$ closest point to x_t in $Y_{\leq t}$ {Ties broken arbitrarily}
  else
    $f_t \leftarrow \perp$
  for $u \in Y_{\leq t}$ **do**
    if $u = f_t$ **then**
      Feed 1 to $\mathcal{A}^u_{\text{count}}$ and let e^u_t be the output
    else
      Feed 0 to $\mathcal{A}^u_{\text{count}}$ and let e^u_t be the output
    Add $(e^u_t - e^u_{t-1})$ copies of u to $\tilde{x}_t$
  return $\tilde{x}_t$

Proof. Fix an adversary $\mathcal{B}$. Recall that we use i^* to denote the challenge index.

First, it will be more convenient to consider a modified algorithm $\widehat{\mathcal{A}}$, which is exactly the same as Algorithm 5, except it releases $(\tilde{x}_t, y_t)$ at each step (instead of just $\tilde{x}_t$). By Lemma 20, it suffice to prove that $\widehat{\mathcal{A}}$ is $(\varepsilon_b + \varepsilon_c, \delta_b)$-DP.

We view $\widehat{\mathcal{A}}$ as the concurrent composition of $\mathcal{A}_{\text{pref}}$ and another mechanism $\widehat{\mathcal{A}}'$ that handles the counters $\{\mathcal{A}^u_{\text{count}}\}_{u \in Y_{\leq n}}$.

- For $\mathcal{A}_{\text{pref}}$, the submitted stream differs only at i^*. Thus, this is $(\varepsilon_b, \delta_b)$-DP.
- For $\widehat{\mathcal{A}}$, conditioned on a transcript prefix $\pi_{t-1} = ((\tilde{x}_1, y_1), \ldots, (\tilde{x}_{t-1}, y_{t-1}))$ and the output y_t from $\mathcal{A}_{\text{pref}}$, $Y_{\leq t} = \bigcup_{t' \leq t} y_{t'}$ agrees in the two runs. Thus, for $t \neq i^*$ the submitted inputs to $\{\mathcal{A}^u_{\text{count}}\}_{u \in Y_{\leq t}}$ agree. For $t = i^*$, assume w.l.o.g. that $x^1_{i^*} = \perp$. For the run $b = 1$, all the inputs to the counters are 0. Meanwhile, for $b = 0$, all the inputs remain 0, except for $\mathcal{A}^{f^0_t}_{\text{count}}$. Hence, we may apply Lemma 22 to conclude that this is ε_c-DP.

Finally, applying Lemma 21 over these two components gives the claimed privacy guarantee. □

Lemma 33 (Efficiency)**.** *Let K be the maximum number of centers produced by $\mathcal{A}_{\text{pref}}$ (i.e., $|Y_{\leq n}| \leq K$). If $\mathcal{A}_{\text{pref}}$ runs in time $\mathcal{T}_b(n)$ and space $\mathcal{S}_b(n)$, and each instance of $\mathcal{A}_{\text{count}}$ runs in time $\mathcal{T}_c(n)$ and space $\mathcal{S}_c(n)$, then Algorithm 5 runs in time $O(\mathcal{T}_b(n) + nKd + K \cdot \mathcal{T}_c(n))$ and uses space $O(\mathcal{S}_b(n) + K \cdot \mathcal{S}_c(n) + Kd)$.*

Proof. Algorithm 5 runs $\mathcal{A}_{\text{pref}}$ across the stream, taking $\mathcal{T}_b(n)$ time and $\mathcal{S}_b(n)$ space. It maintains a set S of at most K points in $\mathbb{R}^d$, requiring $O(Kd)$ space. At each of the n time steps, it computes the distance from x_t to all points currently in S, taking $O(Kd)$ time per step. It then feeds an indicator bit to each of the $|Y_{\leq t}| \leq K$ active instances of $\mathcal{A}_{\text{count}}$. Running K counters takes $O(K \cdot \mathcal{T}_c(n))$ time and $O(K \cdot \mathcal{S}_c(n))$ space over the stream. Summing these bounds yields the stated efficiency. □

C.2 Semi-Coreset Guarantee

Next, we prove that the output of Algorithm 5 provides an output that is a semi-coreset of the input stream. We state the lemma with generic parameters below.

Lemma 34 (Semi-Coreset Guarantee)**.** *Suppose that with high probability, $\mathcal{A}_{\text{pref}}$ provides an (α_1, β_1)-prefix-approximation for all time steps $t \in [n]$, and $\mathcal{A}_{\text{count}}$ is β_2-accurate for all instances and all*

time steps. Then, with high probability, the output multiset stream $\tilde{X} = (\tilde{x}_1, \ldots, \tilde{x}_n)$ *forms a* (κ, η_1, η_2)*-semi-coreset of the input* X *where* $\kappa = O(1+\alpha_1)$*,* $\eta_1 = O(\alpha_1)$*, and* $\eta_2 = O(\beta_1 + K\beta_2)$*, with* $K = |S|$ *representing the total number of centers generated up to time* t*.*

Proof. Fix a time step $t \in [n]$. Algorithm 5 constructs the multiset $\tilde{X}_{\le t}$, which consists of points from $Y_{\le t}$ where the multiplicity of each center $u \in S$ is e_t^u, the estimate provided by the respective instance $\mathcal{A}^u_{\text{count}}$.

We define a natural mapping $\Psi : X_{\le t} \to Y_{\le t}$ that maps each point x_i ($i \le t$) directly to the closest available center $f_i \in Y_{\le i}$ at the time of its arrival: $\Psi(x_i) = f_i$.

Let $c_t^u = \sum_{i=1}^{t} \mathbf{1}[f_i = u]$ be the exact number of points in $X_{\le t}$ that were mapped to center u. The generalized transport cost from the original points $X_{\le t}$ to the estimated coreset $U_{\le t}$ under mapping Ψ is exactly:

$$mt(\Psi, X_{\le t}, \tilde{X}_{\le t}) = \sum_{i=1}^{t} \text{dist}(x_i, f_i)^p + \sum_{u \in Y_{\le t}} |c_t^u - e_t^u|.$$

We bound the two terms separately:

1. **Distance Term:** Since f_i is chosen as the closest point to x_i from the active centers $Y_{\le i}$ output by $\mathcal{A}_{\text{pref}}$, the distance $\text{dist}(x_i, f_i)^p$ is exactly $d(x_i, Y_{\le i})^p$. By the prefix-approximation guarantee of $\mathcal{A}_{\text{pref}}$, we sum this over time to get:
$$\sum_{i=1}^{t} \text{dist}(x_i, f_i)^p \le \alpha_1 \cdot \text{opt}^{k,p}_{X_{\le t}} + \beta_1.$$
2. **Weight Difference Term:** The true frequency of assignment is c_t^u, while the constructed coreset uses the estimated frequency e_t^u. By the assumed accuracy guarantee of $\mathcal{A}_{\text{count}}$, we have $|c_t^u - e_t^u| \le \beta_2$ for all $u \in Y_{\le t}$. Summing this bounded error over all $K = |Y_{\le t}|$ active centers yields:
$$\sum_{u \in Y_{\le t}} |c_t^u - e_t^u| \le K\beta_2.$$

Combining these bounds, the total transport cost strictly satisfies:

$$mt(\Psi, X_{\le t}, U_{\le t}) \le \alpha_1 \text{opt}^{k,p}_{X_{\le t}} + (\beta_1 + K\beta_2).$$

Because the metric space has diameter ≤ 1, we can directly apply Lemma 14 to this result. Substituting $\alpha = \alpha_1$ and $\beta = \beta_1 + K\beta_2$, it follows immediately that $U_{\le t}$ is a valid (κ, η_1, η_2)-semi-coreset for $X_{\le t}$ with parameters $\kappa = O(1+\alpha_1)$, $\eta_1 = O(\alpha_1)$, and $\eta_2 = O(\beta_1 + K\beta_2)$. □

C.3 Putting Things Together: Proof of Theorem 9

Proof of Theorem 9. The algorithm is simply to run Algorithm 5 using $\mathcal{A}_{\text{pref}}$ and $\mathcal{A}_{\text{count}}$ from Theorem 31 and Theorem 23 respectively, where the privacy parameters are set to $\varepsilon_b = \varepsilon_c = \frac{\varepsilon}{2}$ and $\delta_b = \delta$. The privacy guarantee immediately follows from Lemma 32.

Recall also that the size bound by Theorem 31 is $K = O(k^2 \cdot \text{polylog}\,(n/\gamma))$. Plugging this (and the complexity bounds from Theorems 23 and 31) into Lemma 33 yields the claimed time and space complexity.

Using $K = O(k^2 \cdot \text{polylog}\,(n/\gamma))$ and the accuracy guarantees from Theorems 23 and 31 into Lemma 34 yields the semi-coreset guarantee. (Note that the error term is dominated by β_1 in both pure-DP and approximate-DP.) Finally, the number of distinct points in the output is at most K. □

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The introduction contains our main theorem and proof overview with specific forward pointers to relevant sections.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations–including the non-tight dependency on k–are discussed in the conclusion.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All our theoretical results are clearly stated in theorems / lemmas statement with full proofs (either in the main body or in the appendix).

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [N/A]

   Justification: The paper focuses on theoretical results and does not include any experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [N/A]

Justification: The paper focuses on theoretical results and does not include any experiments.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [N/A]

Justification: The paper focuses on theoretical results and does not include any experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [N/A]

Justification: The paper focuses on theoretical results and does not include any experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [N/A]

   Justification: The paper focuses on theoretical results and does not include any experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: -

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: This paper provides differential privacy clustering algorithms–which as discussed in the introduction are crucial in making large-scale data analysis privacy-preserving. We do not foresee any potential negative societal impact of this work.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper focuses on theoretical results and do not involve releasing any new model or data.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [N/A]

    Justification: The paper focuses on theoretical results and does not include any assets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper focuses on theoretical results and does not include any assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper focuses on theoretical results and does not include any experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper focuses on theoretical results and does not include any experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We only use LLM to help with editing the paper, which does not affect any core methods in this research.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.