Differentially-Private Learning of Low Dimensional Manifolds

Anna Choromanska[1], Krzysztof Choromanski[2], Geetha Jagannathan[3], Claire Monteleoni[4]

[1]*Department of Electrical Engineering, Columbia University, NY, USA*
[2]*Department of Industrial Engineering and Operations Research, Columbia University, NY, USA*
[3]*Department of Computer Science, Columbia University, NY, USA*
[4]*Department of Computer Science, George Washington University, NY, USA*
aec2163@columbia.edu, kmc2178@columbia.edu, geetha@cs.columbia.edu, cmontel@gwu.edu

Abstract

In this paper, we study the problem of differentially-private learning of low dimensional manifolds embedded in high dimensional spaces. The problems one faces in learning in high dimensional spaces are compounded in a differentially-private learning. We achieve the dual goals of learning the manifold while maintaining the privacy of the dataset by constructing a differentially-private data structure that adapts to the doubling dimension of the dataset. Our differentially-private manifold learning algorithm extends random projection trees of Dasgupta and Freund. A naive construction of differentially-private random projection trees could involve queries with high global sensitivity that would affect the usefulness of the trees. Instead, we present an alternate way of constructing differentially-private random projection trees that uses low sensitivity queries that are precise enough for learning the low dimensional manifolds. We prove that the size of the tree depends only on the doubling dimension of the dataset and not its extrinsic dimension.

Keywords: differential-privacy, low dimensional manifolds, doubling dimension, random projection tree

1. Introduction

Many real world datasets are measured at an extremely high dimension. Analyzing datasets in the high dimension affects learning algorithms in many ways. Most of the existing algorithms have time complexities that are super-polynomially dependent on the dimension of the dataset. Some algorithms need enormous amounts of data to obtain meaningful results in high-dimensional spaces. This phenomenon is referred to as the curse of dimensionality in the machine learning literature. One way to address this is through dimensionality reduction (Bishop [4], Cox and Cox [13]). In many cases, although a data set may have apparent high dimensionality, the data actually might lie on a low dimensional manifold.

Non-linear dimensionality reduction techniques (Lee and Verleysen [35]) provide ways to construct mappings from the given high dimensional spaces into the low dimensional manifolds in which the data actually lie. Dasgupta and Freund [14] analyzed the technique presented by Freund et al. [23], to learn the structure of a manifold that has low dimension d but for which the data lies in $\mathbb{R}^D$, with $d \ll D$. This involves the construction of a data structure called a random projection tree (RP tree), formed by hierarchically partitioning $\mathbb{R}^D$ into subregions. The height of the RP tree constructed using random projections depends only on the doubling dimension of the dataset. Kpotufe [34] used RP tree to construct a tree based regressor whose convergence rate depends only on the intrinsic dimension of the data.

In this paper, we study the problem of differentially-private learning of low dimensional manifolds. Differential privacy is a privacy model introduced by Dwork et al. [20] in a quest to achieve the dual goal of maximizing data utility and preserving data confidentiality. A differentially-private database access mechanism preserves the privacy of any individual in the database, irrespective of the amount of auxiliary information available to an adversarial database client. The model is described in more detail in Section 4. The problems one faces in learning in high dimensional spaces are compounded in a differentially-private

learning. Differentially private data analysis usually needs more data than its non-private counterpart to achieve a comparable amount of accuracy. It comes from the fact that introduced noise, necessary to achieve desired level of privacy, affects the utility. The amount of data required in high dimensional space for a differentially private learning can make the entire task of constructing a differentially-private algorithm preserving reasonable notion of utility challenging.

2. Our contribution

In this paper, we focus on a data of low doubling dimension as was considered by Dasgupta and Freund [14]. We give a differentially-private manifold learning algorithm that constructs a differentially-private random projection tree that depends only on the doubling dimension of the data. Thus our algorithm extends the random projection tree to the differentially-private setting. A naive way of constructing a differentially-private RP tree would be to replace a non-private data access in the RP tree construction algorithm with an interactive mechanism for differentially-private access to the dataset. However, such a construction involves queries with high global sensitivity (i.e. queries for which there exist datasets that differ on only one point but such that the answers to the queries are arbitrarily far in the corresponding metric space). This results in a substantial reduction in the accuracy of the constructed RP trees. The reason for that is that the non-differentially private algorithm for constructing random projection trees computes the median and this query is highly sensitive. To achieve the desired level of differential privacy in the straightforward approach, a significant noise must be added to each result. That noise dramatically reduces the quality of the constructed random projection tree. We circumvent this issue by constructing a RP tree using low sensitivity queries. We prove that our differentially private RP tree algorithm adapts to the doubling dimension of its input just as the non-private algorithm of Dasgupta and Freund [14]. Our algorithm, as well as the algorithm presented by Dasgupta and Freund [14], is exponential in the doubling dimension d and its sample complexity scales with the square root of the extrinsic dimension D. To the best of our knowledge, this is the first work addressing the curse of dimensionality problem in the differential privacy model using random projection trees. Our work is theoretical and we do not optimize constants appearing in the algorithm. However, we emphasize that with more calculations (tedious but not hard) most of them may be significantly improved to make the algorithm applicable in the real-life scenarios.

3. Related Work

The desire to maximize data utility while preserving the confidentiality of individuals in a database has led to the proposal of a number of privacy models including perturbation methods (Adam and Worthmann [1], Agrawal and Srikant [2]), k-anonymity and its variants (Samarati and Sweeney [40], Sweeney [41]) and secure multiparty computation (Goldreich [26], Lindell and Pinkas [36]). The weakness of these privacy models have also been well studied (Ganta et al. [25], Brickell and Shmatikov [7]). The differential privacy framework introduced by Dwork et al. [20] offers strong privacy guarantees for every individual in the database irrespective of any auxiliary information that is available to the database client. Following the work of Dwork et al. [20] significant amount of work has been done in this area and most of them are surveyed by Dwork [15, 16, 17, 18].

Learning algorithms have also been studied under the differential privacy model. Various private data mining algorithms such as PCA, k-means clustering, ID3 are presented in a privacy model called SuLQ (Blum et al. [5]), which is a predecessor of differential privacy. Ignoring computational constraints, Kasiviswanathan et al. [33] showed that anything which is PAC-learnable is also differentially-private PAC-learnable. Building upon their technique, Blum et al. [6] showed a way of constructing a synthetic database useful in any concept class with polynomial VC-dimension. Their construction is computationally inefficient. Chaudhuri et al. [9] showed that it is possible to obtain differentially-private empirical risk minimization algorithms by perturbing their objective functions. Feldman et al. [21] gave an algorithm for computing differentially private coresets that could answer k-median and k-mean queries in $\mathbb{R}^d$. The size of the released dataset is unreasonably large for most values of d. Jagannathan et al. [29] presented a differentially private classifier based on random

decision trees. Their algorithm achieves good accuracy even for small datasets. Friedman and Schuster [24] presented a differentially private ID3 algorithm that gives better accuracies than the straightforward construction of a differentially private ID3 tree. Recently, Chaudhuri and Hsu [8] analyzed the sample complexity bounds for differentially-private learning. To the best of our knowledge our paper is the first one that addresses the problem of the construction of differentially-private random projection trees.

Algorithm 1: Algorithm **MakeTree** (**RPtree**)

Input: Dataset X.
Output: A decision tree.
begin
  if $|X| < MinSize$ **then**
    return $(Leaf)$
  else
    $Rule \leftarrow$**ChooseRule**(X);
    $LeftTree \leftarrow$**MakeTree**$(\{x \in X : Rule(x) = \text{true}\})$;
    $RightTree \leftarrow$**MakeTree**$(\{x \in X : Rule(x) = \text{false}\})$;
    return $([Rule, LeftTree, RightTree])$;
  end

  Subroutine **ChooseRule**(X)
    choose a random unit direction $v \in \mathcal{R}^D$;
    pick any point $x \in X$, and let y be the farthest point from it in X;
    choose s uniformly at random in $[-1, 1] \cdot 6||x - y||/\sqrt{D}$;
    $Rule(x) := x \cdot v \leq (\text{median}(\{z \cdot v : z \in X\}) + s)$;
    return $(Rule)$;
end

However there are several papers where differentially-private constructions of other important structures are presented. Cormode et al. [11] consider the problem of differentially private release of a sparse data. Chaudhuri et al. [10] investigated the performance of a differentially private principal component analysis which is used for dimensionality reduction. Finally, very recently Kapralov and Talwar [32] presented an algorithm that outputs a differentially-private approximation to the principal eigenvector of a given symmetric matrix. Differential privacy has also been studied in the online learning context (see for example: [30]). Another interesting setting involves scenario, where the access to the training features is only through a kernel function (see: [31]).

4. Preliminaries

Differential privacy is a model of privacy for database access mechanisms. It captures a notion of individual privacy by assuring that the removal or addition of a single item (i.e., an individual's record) in a database does not have a substantial impact on the output produced by the mechanism. Two databases D_1 and D_2 *differ on at most one element* if one is a proper subset of the other and the larger database just contains one additional row.

Definition
4.1 (Dwork et al. [20])**.** *A randomized algorithm $\mathcal{M}$ satisfies* ϵ-differential privacy *if for all databases D_1 and D_2 differing on at most one element, and all $S \in Range(\mathcal{M})$, $Pr[\mathcal{M}(D_1) = S] \leq \exp(\epsilon) \cdot Pr[\mathcal{M}(D_2) = S]$. The probability is taken over the coin tosses of $\mathcal{M}$.*

Smaller values of ϵ correspond to closer distributions, and therefore higher levels of privacy. Let f be a function on databases with range $\mathbb{R}^m$. A now-standard technique by which a mechanism $\mathcal{M}$ that computes a noisy version of f over a database X can satisfy *ϵ-differential privacy* is to add noise from a suitably chosen distribution to the output $f(X)$. The magnitude of the noise added to the output depends on how much change in f, can be caused by a single change to the database, defined as follows:

Definition
4.2 (Dwork et al. [20]). *The* global sensitivity *of a function f is the smallest number $S(f)$ such that for all D_1 and D_2 which differ on at most one element, $\| f(D_1) - f(D_2) \|_1 \leq S(f)$.*

Let $\text{Lap}(0, \lambda)$ denote the Laplace distribution with mean 0 and standard deviation λ. Throughout the paper λ denotes the differentially-private parameter.

Theorem
4.1 (Dwork et al. [20]). *Let f be a function on databases with range $\mathbb{R}^m$. Then, the mechanism that outputs $f(X) + (Y_1, \ldots, Y_m)$, where Y_i are drawn i.i.d from $\text{Lap}(0, S(f)/\epsilon)$, satisfies ϵ-differential privacy.*

Using this method, smaller values of ϵ imply that more noise is added when the results are returned. The following theorem shows that differential privacy is robust under composition, but with an additional loss of privacy for each query made.

Theorem
4.2 ((Dwork et al., 2006)). ***(Composition Theorem)*** *The sequential application of mechanisms $\mathcal{M}_i$, each giving ϵ_i-differential privacy, satisfies $\sum_i \epsilon_i$-differential privacy.*

We first introduce some notation we will be using throughout the paper. Let $X \subseteq \mathbb{R}^D$ be the dataset on which differentially-private RP tree is built. Let us assume that X lies within a hypercube with the center at $(0, 0, \ldots, 0)$ and side length ℓ. We assume ℓ is public. For any point $x \in \mathbb{R}^D$ and any $r > 0$, let $B(x, r) = \{z : \|x - z\| < r\}$ denote the open ball of radius r centered at x. The radius of a cell $A \subset \mathbb{R}^D$ is the smallest $r > 0$ such that $X \cap A \subset B(x, r)$ for some $x \in A$ or $x \in X$. We denote *diam*(A) to be the diameter of A which is twice the radius r. Let I be an interval that is divided into n equal subintervals $I_1, \ldots, I_n$. We denote $I = I_1 \cup \ldots \cup I_n$ as $I = I_1 ... I_n$.

5. Random Projection Trees: An Overview

A random projection tree (Dasgupta and Freund [14]) is a variant of a *k-d* tree. *k-d* trees partition the space $\mathbb{R}^D$ into hyperrectangular cells by splitting along one coordinate at each node of the tree. Although simple in construction, they suffer from the "curse of dimensionality," as do many nonparametric statistical methods. The trees become less useful as the dimension, D, increases. Dasgupta and Freund [14] showed that there is a dataset in $\mathbb{R}^D$ for which a *k-d* tree requires D levels in order to halve the cell diameter.

However, there are many datasets that appear to lie in very high dimensional space, but actually lie in a low dimensional manifold. In order to address this situation, Dasgupta and Freund [14] gave a variant of the *k-d* tree named *random projection tree* that adapts to the low dimensional structure of the dataset without having to explicitly learn the structure. The random projection tree is also a spatial data structure built by recursively splitting the data space. At each node of the tree, a direction is chosen at random from a unit square in $\mathbb{R}^D$ and the subset of data points at each node are projected on the chosen random direction. Instead of choosing the median of these projected points as the split, the RP tree algorithm involves adding a small amount of "jitter" and the split point is chosen at random from the jitter centered at the median. Algorithm 1 shows the construction of the random projection tree as given by Dasgupta and Freund [14]. They proved a bound on the rate at which the radius of cells in an RP tree decreases as one moves down the tree.

Definition
5.1. *The doubling (or Assouad) dimension of the set $S \subset \mathbb{R}^D$ is the smallest integer d such that for any ball $B(x, r) \subset \mathbb{R}^D$, the set $B(x, r) \cap S$ can be covered by 2^d balls of radius $r/2$.*

Algorithm 2: Algorithm **MakeDPRPTree** (**DP-RPtree**)

Input: Dataset X, the height of the current node h, constants K and $g < 1/2$.
Output: A decision tree.
begin
  if $h > MaxTreeHeight$ **then**
    return $(Leaf)$
  else
    $Rule \leftarrow$**ChooseDPRule**(X);
    $LeftTree \leftarrow$**MakeDPRPTeee**$(\{x \in X : Rule(x) = \text{true}, h+1\})$;
    $RightTree \leftarrow$**MakeDPRPTeee**$(\{x \in X : Rule(x) = \text{false}, h+1\})$;
    return $([Rule, LeftTree, RightTree])$;
  end

  Subroutine **ChooseDPRule**(X)
    $radius \leftarrow$**ChooseRadius**(X);
    $R \leftarrow K \cdot radius$;
    choose random direction U;
    $M \leftarrow$**ComputeMedian**(X, U);
    choose s uniformly at random in $[-6.6, 6.6] \cdot 2 \cdot R$;
    return $(Rule(x) := x \cdot U \leq M + s)$;

  Subroutine **ChooseRadius**(X)
    $radius \leftarrow 0$;
    for $i \in \{1, \ldots, 14\}$;
      choose random direction U;
      M $\leftarrow$ **ComputeMedian**(S, U);
      find $R = \min\{m\delta : [M - m\delta, M + m\delta]$ is g-good$\}$;
      if $radius < R$ **then**
        $radius \leftarrow R$;
    return radius;

  Subroutine **ComputeMedian**(X, U)
    partition $[-L/2, L/2]$ into n segments;
    let $I_1, \ldots, I_n$ denote the segments and $m(i) = |\{x_i \in X : U \cdot x \in I_i\}|$;
    choose $p(i) \sim \text{Lap}(0, 1/\lambda)$ and $N \leftarrow \sum_{i=1}^{n}(m(i) + p(i))$;
    if $N < 0$ **then**
      pick a $1 \leq j \leq n$ uniformly at random;
    else
      find j such that $\sum_{i=1}^{j-1}(m(i) + p(i)) \leq \frac{1}{2}N$ and $\sum_{i=1}^{j}(m(i) + p(i)) \geq \frac{1}{2}N$;
    return left end point of I_j;
end

The following theorem is the main ingredient of the proof that random projection tree constructed in Dasgupta and Freund [14] is of good quality. We will obtain similar result for a differentially-private random projection tree constructed by us in this paper.

Theorem
5.1 (Dasgupta and Freund [14])**.** *There is a constant c with the following property: Suppose an RP tree is built using the dataset $X \in \mathcal{R}^D$. Pick any cell C in the RP tree. Suppose $X \cap C$ has doubling (or Assouad) dimension d. Then, with probability at least $1/2$, for every descendant C' which is more than $c \log d$ levels*

below C, we have $radius(C') < radius(C)$.

6. Differentially-Private Random Projection Tree

In this section, we describe our algorithm for constructing a differentially-private RP tree. Our algorithm (Algorithm 2) is a non-trivial modification of the RP tree algorithm given by Dasgupta and Freund [14]. We will start from presenting some definitions that are relevant to our construction of the differentially-private RP tree and in proving that our differentially-private RP tree adapts to the doubling dimension.

Definition
6.1. *A set X of N points and diameter Δ is (η, W)-dense if at least $(1-\eta)N$ points of X are within a ball of radius $\frac{\Delta}{W}$.*

Definition
6.2. *Consider a sequence of real numbers $(\beta_1, ..., \beta_n)$ where each β_i is associated with an interval I_i and let $N_p = \sum_{i=1}^{n} \beta_i$. For a constant $0 < g < 1$, we say that the interval $I = I_1, I_2, ...I_t$ is g-good if the following holds: $\sum_{i=1}^{t} \beta_i \geq (1-g)N_p$.*

Definition
6.3. *A set X of diameter Δ and doubling dimension d is (T, ρ, f)-good if there exist two balls $B_1(c_1, r_1)$ and $B_2(c_2, r_2)$ with radii $r_1, r_2 < \frac{w}{\rho\sqrt{d}}$ and $|c_1 - c_2| = w \geq \frac{\Delta}{T}$, such that each of the balls contains at least $f|X|$ of all the points from X.*

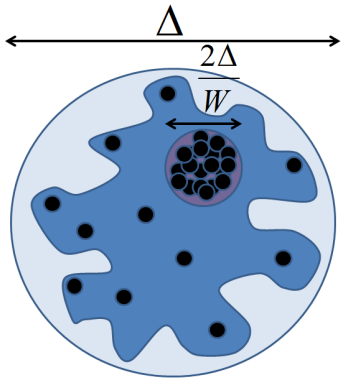

Figure 1: (η, W)-dense set of points X. Most of the point lie within radius $\frac{\Delta}{W}$ from the center of the small ball in the picture. However the diameter of the set Δ is much larger ($W \gg 1$).

Before describing our algorithm, we will first explain a naive conversion of the RP tree algorithm of Dasgupta and Freund [14] into a differential privacy-preserving mechanism that does not yield trees that are good representatives of the dataset. A straightforward way of constructing a differentially-private RP tree is by replacing each database access in the non-private algorithm with a differentially private query. By applying Theorem 4.2, one can show that such a construction is differentially-private. The following computations in the procedure **ChooseRule**(X) in Algorithm 1 require an access to the database:

- *pick any point $x \in X$ and compute the distance between the point x to the farthest point $y \in X$.* This distance is used to estimate the data diameter of X.
- $Rule(x) := x \cdot v \leq (\text{median}(\{z \cdot v : z \in X\}) + s)$ where s is chosen uniformly at random.

Let us comment more on some alternative possible approaches. The straightforward way to extend the results of Dasgupta and Freund [14] is to replace both the diameter and the median computations with the

differentially-private diameter and median computations (Dwork et al. [20], Beimel et al. [3]). However, both the median and the diameter have high global sensitivity. Thus the direct way of doing this: computing the median and the diameter of the input dataset S, then adding an appropriate noise term chosen from a Laplace distribution (with a parameter that is linearly dependent on the global sensitivity of these queries) substantially impacts the precision of the differentially-private median and diameter computations. Hence, the produced random projection tree is of very low utility. There are several other variants of computing the approximate median but none of them matches well our setting. Let us list the most important of them now. The first alternative approach we will discuss is the exponential mechanism (McSherry and Talwar [37]) that provides a way to compute the differentially-private median. The importance of the median comes from the fact that it provides a good split and consequently the diameter of data stored in the node decreases quickly along the path from the root to the leaf. This is the case for the approximated median computation we propose in this paper. This is however not the case when the exponential mechanism is used. The latter one provides a good split with constant probability only if the data is not skewed (Cormode et al. [12]). We do not need this assumption in our setting and in fact in all real life applications data is very often skewed. Another important mechanism that can be potentially applied here is the so-called smooth sensitivity (see for instance: the Propose-Test-Release mechanism provided by Dwork and Lei [19]). However, the differentially private median computed using smooth sensitivity (Nissim et al. [38]) offers a weaker privacy guarantee ((ϵ, δ)-differential privacy) than the standard differential privacy model. The guarantees we obtain are standard differential privacy guarantees. What should be also emphasized is that it is unclear how any of these mechanisms, when used directly in the algorithm, can be proved to provide useful upper bounds on the depth of the differentially-private random projection tree. Controlling the depth of the tree is the key ingredient that is needed to prove any results regarding the utility of the obtained random projection tree.

Computing median in the setting where privacy guarantees are required can be used with the use of the cell-based methods (see for instance: Inan et al. [28]). In that methods very often the median query is replaced with multiple count queries, one for each value within the particular range. The median is then calculated based on the noisy counts. Even though direct median query is very sensitive, that approach has fixed sensitivity, becaue query regions are disjoint. In this paper we present a procedure to compute approximations to the median using only low sensitivity count queries (thus in this sense our approach is similar to the one presented in Inan et al. [28]). We use a much weaker assumption on the density of the data though, namely, that the data is not entirely concentrated in a very small neighborhood. Note that even though our algorithm divides intervals into equal sized segments, the analysis makes no additional assumptions regarding the distribution of the data. In Blum et al. [6] the authors proposed a scheme of constructing a synthetic database useful for any concept class with polynomial VC-dimension. Their approach is however not computationally efficient.

To sum it up, our algorithm is conceptually much simpler than most mentioned approaches, yet it outperforms then in terms of privacy guarantees, running time or assumptions about data distribution (our assumptions are much more general than most requirements needed in the alternative approaches). Furthermore, it can be easily implemented. Finally, this is another approach (different from all mentioned before) that gives strong privacy guarantees and at the same time provides practically useful random projection trees.

Let us mention some other work regarding median computation in the
differentially-private or related privacy settings. Several optimization problems regarding computing medians in the differentially-private way were analyzed (see for istance: Gupta et al. [27]). One of the most important problems here is the so-called *k-median problem*. We are given a set of points V and a metric space on them: $dV \times V \to R$. The private set of demand points $D \subseteq V$ is also given. The goal is to select a set of medians $F \subseteq V$ with $|F| = k$ to minimize the quantity $c(F) = \sum_{v \in D} d(v, F)$, where: $d(v, F) = \min_{f \in F} d(v, f)$. This problem was considered for example in Gupta et al. [27] but also independently in Feldman et al. [22]. The two models are slightly different. The result from Gupta et al. [27] gives smaller additive errors and outputs a k-median approximation. In the one from Feldman et al. [22] the coresets for the problem are output. The lower bound argument for private coresets is similar to the one presented in Gupta et al. [27]. We should notice that tree-based partitioning techniques are extremely important in the differentially-private context.

Some methods involve private spatial decomposition. Others focus on data-dependent structures, where a privacy mechanism is needed for choosing splitting points. This is exactly our setting, where we need a private median construction. In several approaches a nested tree structure with consistency enforcement and adaptive privacy budget assignment (see [39]) is built . Those results usually improve the asymptotic error bound and query accuracy compared to the earlier methods.

In our construction, we avoid high-sensitivity queries by computing approximations to the median and the diameter that are precise enough for our purposes of finding a low-dimensional manifold. We describe below the two procedures that involve computing the approximate median and the approximate data diameter. The differentially-private random projection tree construction is given in Algorithm 2. We denote this tree as **DP-RPtree**. The algorithm is parametrized by scalars g and K. The appropriate way of choosing them will be explained later in the paper.

6.1. Computing an approximate median

We assume that all data is taken from the D-dimensional box with center $(0, 0, ..., 0)$ and edges of length ℓ, where ℓ is public. Let C denote the cell that needs to be partitioned. Let $X \cap C$ be the set of data points in the cell C. First, we choose a random unit vector $U \in \mathbb{R}^D$. The projection of any data point in $X \cap C$ onto the unit vector U lies in the segment $[-\frac{L}{2}, \frac{L}{2}]$ where $L = 2\sqrt{(\ell/2)^2 D}$. Since both ℓ and D are public, L is also public. We partition the segment $[-\frac{L}{2}, \frac{L}{2}]$ into n subsegments, $I_i, 1 \leq i \leq n$. Denote the length of each subsegment by δ (we have: $\delta = \frac{L}{n}$). We call δ a *precision parameter*. Both n and δ are public. We compute differentially private counts of the projected points β_i that fall into these subregions I_i. We use these counts to estimate an approximate median of the projection of $X \cap C$. The approximate median is defined as the left end of the subsegment I_j where $\sum_{i=1}^{j-1} \beta_i \leq \frac{1}{2} \sum_{i=1}^{n} (\beta_i)$ but $\sum_{i=1}^{j} \beta_i \geq \frac{1}{2} \sum_{i=1}^{n} \beta_i$ for $N_p = \sum_{i=1}^{n} \beta_i \geq 0$. For $N_p \leq 0$ the median is defined as an arbitrary endpoint of an arbitrary subsegment I_i. In other words, the left endpoint of the lowest numbered subregion for which at least half the projected points lie to the left is our approximate median. Since our application uses noisy counts of projected points, β_i can be negative for some i. Hence, the perturbed median is not uniquely defined even for $N_p > 0$. However there exists at least one median. This is described in procedure **ComputeMedian**(X) in Algorithm 2. We refer to the median defined above as the *perturbed median*.

6.2. Computing an approximate diameter

Now, we describe briefly the procedure that computes an approximate data diameter. Here we choose 14 unit vectors $\{U_1, \ldots, U_{14}\} \subset \mathbb{R}^D$ at random from the Gaussian distribution. It turns out that to prove the correctness of our algorithm we indeed need at least 14 Gaussian vectors. Since this is due to some highly technical reasons and we now try to provide a general view of our algorithm, we will not explain it more exhaustively at this moment but it will be justified later in the paper. We first compute the approximate medians projected onto the chosen vectors. For each U_i we find the smallest portion of the segment $[-\frac{L}{2}, \frac{L}{2}]$ around the median such that the subsegment is g-good for a suitable parameter $0 < g \leq 1$. Parameter g should be small enough for the algorithm to work (how small it should will be shown later). Thus in every trial we find some subsegment. We choose the longest subsegment among 14 that were found. This is described in procedure **ChooseRadius**(S) in Algorithm 2.

The following theorem shows that random projection tree constructed by our algorithm is differentially private. We assume that the tree is of height h.

Theorem
6.1. DP-RPtree *is* $14h\lambda$ *differentially-private.*

Proof. Let A denote the differentially-private random projection tree algorithm. To construct a random projection tree of height h we need $14h$ queries to the private data. Each time we obtain a vector of differentially-private counts which is of global sensitivity 1 (according to Theorem 4.1). Therefore, using the Composition Theorem, one can now show that **DP-RPtree** is $14h\lambda$-differentially-private.

□

Thus we only need to analyze the quality of the random projection tree constructed by our algorithm. We will focus on that in the remaining part of the paper.

7. Differentially-Private RP Trees Adapt to Doubling Dimension

In this section we state and prove our main result, that the height of the differentially-private random projection tree depends only on the doubling dimension of the dataset and the privacy parameter. This section is organized as follows: first we state the main theorem (Theorem 7.1) and also a slightly stronger result (Theorem 7.2) that implies the main theorem, then we we provide a brief outline of the proof and finally we show all the lemmas and technical proofs that led to the presented results.

7.1. Main Theorem

Let $X \subseteq \mathbb{R}^D$ be the dataset of doubling dimension d on which the **DP-RPtree** is built. Let A denote a cell of the RP tree. By ρ_A we denote the average density of A (i.e the ratio of the number of data points in A over the volume of A). We prove that, with high probability, every descendant of a cell C which is $O(d \log d)$ levels below has half the radius of C.

We first introduce a set of assumptions which will be used in the following theorems.

Assumption A1 Let $X \subseteq R^D$ and let A be any cell of the differentially-private RP tree. Assume the following:

(a) $X \cap A$ has doubling dimension $\leq d$.

(b) $X \cap A$ has diameter 2Δ.

(c) $X \cap A$ contains N points.

(d) There is no ball of a positive radius in $X \cap A$ whose density is greater than $\frac{W^D}{2}\rho_A$, where W is a positive constant.

Theorem

7.1. *Let $X \subseteq R^D$. Also let W be a positive constant, $K = 400W$, and $N = \Omega(\frac{e^{2d} d^{\frac{d}{2}} n \log^2(n)}{\lambda})$ for $n = \frac{L}{\delta}$. Pick any cell A of the differentially-private RP tree, and let assumption A1 hold. Assume that algorithm parameters g and δ are small enough. Then the probability that there exists a descendant of A which is more than $\Omega(d \log(d))$ levels below and has radius at least $\frac{\Delta}{2}$ is at most $\frac{1}{2}$.*

This theorem shows that our algorithm achieves a similar reduction in size of the data diameter of the cell, as was achieved by Dasgupta and Freund [14] while preserving differential privacy. Note that the smaller the precision parameter δ, the bigger n and thus we need more data points in the theorem. This agrees with our inuition since smaller length δ of the subsegment affects the privacy guarantees and therefore to obtain the same type of differential-privacy we need more points.

In fact, we prove slightly stronger result than Theorem 7.1. This result is stated in Theorem 7.2 that we now provide. Let $\nu(x) = \frac{(\lambda x)^n}{\exp(\lambda x)}$ and ψ_λ^n is the inverse of $\nu(x)$, defined on $[\frac{n}{\lambda}, \infty]$. Before showing the theorem, we first introduce the following notation: $\zeta_1 = \frac{20n}{3\lambda}$, $\zeta_2 = \frac{8n}{f\lambda}$, $\zeta_3 = \frac{4n}{(1-f)\lambda}$, $\zeta_4 = \frac{16}{\lambda f}$, $\zeta_5 = \psi_\lambda^n(\frac{5}{3ne^{15}})$, $\zeta_6 = \psi_\lambda^n(\frac{8}{fn^2e^{31}})$, $\zeta_7 = \psi_\lambda^n(\frac{8}{fe^{15}})$, $\zeta_8 = \psi_\lambda^n(\frac{1}{(1-f)ne^{15}}))$, $n = L/\delta$ and $f = \frac{\eta}{CM^d}$ for $M = 130e^2\sqrt{d}$ and some constant C. Furthermore, we define the set of following assumptions regarding constants δ, g, K from Algorithm 2.

Assumption A2 Assume that constants δ, g, K from Algorithm 2 satisfy:

(a) $g < \frac{\eta}{2C}(\frac{1}{130e^2T\sqrt{d}})^d$

(b) $\zeta_9 > K > 2e^2T$ for $\zeta_9 = \frac{0.00094e^{31}}{18(\sqrt{30+2\log(\frac{2C}{\eta})+2d\log(130e^2T\sqrt{d})}+2\delta)}$

(c) $\delta \leq \frac{0.1\Delta}{\sqrt{D}}$,

where $0 \leq \eta \leq 1$, $T = (1 + \frac{2e^2}{\sqrt{d}})W$, and W is some constant.

Theorem
7.2. *Let $X \subseteq R^D$. Pick any cell A of the differentially-private RP tree, and let assumption A1 (a-c) hold. Assume the set $X \cap A$ is not (η, W)-dense for some constant W, where $0 \leq \eta \leq 1$ and $N > \max(\zeta_1, \zeta_2, \zeta_3, \zeta_4, \zeta_5, \zeta_6, \zeta_7, \zeta_8)$, and assume A2 holds. Then the probability that there exists a descendant of A which is more than $\Omega(d \log(d))$ levels below and has radius at least $\frac{\Delta}{2}$ is at most $\frac{1}{2}$.*

Theorem 7.2 implies Theorem 7.1 as follows: If $X \cap A$ is (η, W)-dense then there exists a ball B of radius $\frac{\Delta}{W}$ that contains all but at most ηN points of the data. So the average density inside this ball is at least $\frac{W^D}{2}\rho_A$. Taking $\eta = \frac{1}{2}$ and simplifying the lower bound on the number of data points N, we prove Theorem 7.1.

We give here a brief outline of the proof of Theorem 7.2 (the formal proof can be found in the Appendix). We cover $X \cap A$ by $N_b = O(\sqrt{d}^d)$ balls each of radius $\Delta/\sqrt{d}$. We prove that if we pick any two balls from N_b that are separated by a distance of at least $(\Delta/2) - (\Delta/(512C\sqrt{d}))$ then with constant probability a split point carefully chosen using a constant number of random projections separates the two balls. We also show that any pair of balls that are separated by a distance of at least $(\Delta/2) - (\Delta/(512C\sqrt{d}))$ are separated after $O(d \log d)$ levels with probability at least 1/2. Hence each cell contains points that are within a distance $(\Delta/2) - (\Delta/(512C\sqrt{d}))$ of each other thus proving that the radius$(X \cap A) \leq \Delta/2$.

Although superficially the outline of our proof looks similar to the one in the work of Dasgupta and Freund [14], we emphasize that both proofs substantially differ in details. This is because our **DP-RPtree** construction satisfies the dual constraints of privacy and adaptation to the doubling dimension of the dataset. Our tree construction, unlike the one proposed by Dasgupta and Freund [14], uses approximate median and diameter. The difficulty lies in proving that the approximate median and the diameter used in the construction of the **DP-RPtree** are precise enough to learn the structure of the low dimensional manifold.

8. Appendix - Proof of the Main Theorem

To prove Theorem 7.2, we need to first show the properties of the random projections, properties of the perturbed median and properties of the split.

Properties of the random projections

Random projections are chosen from a multivariate Gaussian distribution $U \sim N(0, \frac{1}{D} I_D)$. Dasgupta and Freund [14] proved certain properties of random projections and showed how they affect the data diameter. Below we list those properties as Lemmas 8.1, 8.2 and 8.3. Their proofs can be found in the work of Dasgupta and Freund [14].

Lemma
8.1. *[Dasgupta and Freund [14]]*
Fix any $x \in R^D$. Pick a random vector $U \sim N(0, \frac{1}{D} I_D)$. Then for any $\alpha, \beta > 0$ the following holds:

- $P[|U \cdot x| \leq \alpha \frac{|x|}{\sqrt{D}}] \leq \sqrt{\frac{2}{\Pi}}\alpha$
- $P[|U \cdot x| \geq \beta \frac{|x|}{\sqrt{D}}] \geq \frac{2}{\beta} e^{-\frac{\beta^2}{2}}$,

where $|x|$ is an euclidean norm of vector x.

Thus this lemma states that a random projection $\mathbb{R}^D \to \mathbb{R}$ approximately preserves the lengths of the vectors, modulo a scaling factor of $\sqrt{D}$.

Lemma
8.2. *[Dasgupta and Freund [14]] Suppose that $X \subseteq R^D$ lies within ball $B(x_0, \Delta)$. Pick any $0 < \delta, \epsilon \leq 1$ such that $\delta\epsilon \leq \frac{1}{e^2}$. Denote by μ some probabilistic measure on X. Then with probability at least $1 - \delta$ over the choice of random projection onto R, all but an ϵ fraction of $\hat{X}$ (measured according to μ) lies within distance $\sqrt{2\log(\frac{1}{\delta\epsilon})}\frac{\Delta}{\sqrt{D}}$ of $\hat{x_0}$.*

Thus this lemma states that the median of the projected points lie in that *central interval*.

Lemma
8.3. *[Dasgupta and Freund [14]] Suppose set $X \subseteq R^D$ is contained in a ball $B(x_0, \Delta)$ and has doubling dimension d. Denote by $\hat{X}$ random projection of X into R. Then for any $0 < \delta < 1$ with probability greater than $1-\delta$ over the choice of projection, $\hat{X}$ is contained in an interval of radius $4\frac{\Delta}{\sqrt{D}}\sqrt{2(d + \log(\frac{2}{\delta}))}$ centered at $\hat{x_0}$.*

Thus this lemma states that a set $X \subset \mathbb{R}^D$ of radius Δ whose doubling dimension is d projects to a *central interval* in $\mathbb{R}$ of radius at most $O(\Delta/\sqrt{D/d})$.

Properties of the Perturbed Median

In order to compute an approximate median, we use different technique than Dasgupta and Freund [14]. In our paper, we are required to compute a differentially-private median. As explained in Section 6, instead of computing a differentially-private median in a traditional way which involves computing the median and then adding appropriate Laplacian noise, we compute approximations to the medians. Below we first provide the list of useful properties of the approximate medians. Those properties follow from the lemmas that are given below.

1. If an interval I in the projected line contains only a small fraction of projected points of X, then I also contains only a small fraction of differentially-private count of projected points and vice-versa (consequence of Lemmas 8.4, 8.5 and 8.6).
2. The perturbed median M lies close to $\hat{x_0}$, where $\hat{x_0}$ is the projection of the center of the ball $B(x_0, \Delta)$ (consequence of Lemma 8.7).
3. If an interval contains $1 - g$ fraction of projected points, then the perturbed median lies within that interval with high probability (consequence of Lemma 8.8).

The above properties are the direct consequences of the technical lemmas that we will provide now.

Lemma
8.4. *Let $\{l_1, ..., l_n\}$ be a family of independent Laplace random variables $L(0, 1/\lambda)$. Then for any $W > \frac{2}{\lambda}$ we have: $P[l_1 + ... + l_n \geq W] \leq \frac{(\lambda W)^n}{e^{\lambda W}}$.*

Proof. This lemma proves an upper bound for the tail of the sum of independent Laplace random variables. Denote $Y = l_1 + ... + l_n$. We will use the following formula for the density g_Y of Y:

$$g_Y(y) = \sum_{k=0}^{n-1} \frac{(n+k-1)!}{k!(n-1)!} \frac{1}{2^{n+k}} \chi(y, n, k, \lambda),$$

where: $\chi(y, n, k, \lambda) = \frac{\lambda^{n-k}}{(n-k-1)!} e^{-\lambda|y|} |y|^{n-k-1}$.

So we have:

$$P[Y \geq W] = \sum_{k=0}^{n-1} \frac{(n+k-1)!}{k!(n-1)!} \frac{1}{2^{n+k}} \frac{\lambda^{n-k}}{(n-k-1)!} \int_W^\infty e^{-\lambda y} y^{n-k-1} dy \tag{1}$$

Denote $T(m) = \int_W^\infty e^{-\lambda y} y^m dy$. By the differentiation by parts we easily obtain the following recursive formula:

$$T(m) = \frac{1}{\lambda}(e^{-\lambda W} W^m + mT(m-1)) \tag{2}$$

So we obtain:

$$T(m) = \frac{e^{-\lambda W}}{\lambda}(W^m + \frac{m}{\lambda}W^{m-1} + \frac{m(m-1)}{\lambda^2}W^{m-2} + ... + \frac{m!}{\lambda^m}) \tag{3}$$

We have:

$$P[Y \geq W] = \sum_{k=0}^{n-1} \frac{(n+k-1)!}{k!(n-1)!}\frac{1}{2^{n+k}}\frac{\lambda^{n-k}}{(n-k-1)!}T(n-k-1), \tag{4}$$

so

$$P[Y \geq W] = \sum_{k=0}^{n-1} \frac{(n+k-1)!}{k!(n-1)!}\frac{1}{2^{n+k}}\frac{\lambda^{n-k}}{(n-k-1)!}\frac{e^{-\lambda W}}{\lambda}L(n,k), \tag{5}$$

where:

$$L(n,k) = \sum_{i=0}^{n-k-1} \frac{1}{i!}(n-k-1)!\frac{W^i}{\lambda^{n-k-1-i}} \tag{6}$$

So we obtain:

$$P[Y \geq W] = e^{-\lambda W}\sum_{k=0}^{n-1} \frac{(n+k-1)!}{k!(n-1)!}\frac{1}{2^{n+k}}\sum_{i=0}^{n-k-1}\frac{(\lambda W)^i}{i!} \tag{7}$$

We have:

$$\sum_{i=0}^{n-k-1}\frac{(\lambda W)^i}{i!} \leq \sum_{i=0}^{n-k-1}(\lambda W)^i = \frac{1-(\lambda W)^{n-k}}{1-\lambda W} \leq (\lambda W)^{n-k} \tag{8}$$

for $W > \frac{2}{\lambda}$.
Therefore we have:

$$P[W \leq Y] \leq e^{-\lambda W}\sum_{k=0}^{n-1} 2^{n+k-1}\frac{1}{2^{n+k}}(\lambda W)^{n-k} = \frac{e^{-\lambda W}}{2}\sum_{k=0}^{n-1}(\lambda W)^{n-k} \tag{9}$$

So:

$$P[W \leq Y] \leq \frac{e^{-\lambda W}}{2}\frac{\lambda W}{\lambda W - 1}((\lambda W)^n - 1) \leq \frac{(\lambda W)^n}{e^{\lambda W}} \tag{10}$$

for $W \geq \frac{2}{\lambda}$.
So $P[Y \geq W] \leq \frac{(\lambda W)^n}{e^{\lambda W}}$ for $W > \frac{2}{\lambda}$.

□

Lemma
8.5. *Fix some constants $0 < h, h^{'} < 1$ such that $h + h^{'} < 1$. Fix some interval $Int = I_{j+1}, ..., I_{j+t}$ for some j, t, where $t \geq 1$ that contains at most a fraction h of all N projected points, where $N > \frac{4}{\lambda h^{'}}$. Then with probability at most $(\nu(\frac{h^{'}N}{2}) + \nu(\frac{h^{'}N}{2(h+h^{'})}))$ we have:*

$$\sum_{i=1}^{t}(m(j+i) + p(j+i)) \geq (h+h^{'})\sum_{i=1}^{n}(m(i) + p(i)),$$

where each $p(i) \sim L(0, 1/\lambda)$.

Proof. Assume w.l.g. that $Int = I_1, \ldots, I_t$. Assume that $\sum_{i=1}^{t}(m(i)+p(i)) \geq (h+h^{'})\sum_{i=1}^{n}(m(i)+p(i))$. Using the fact that Int contains at most hN projected points, we obtain

$$\sum_{i=1}^{t} p(i) - (h+h^{'})\sum_{i=1}^{n} p(i) \geq h^{'}N \tag{11}$$

So we have either

$$\sum_{i=1}^{t} p(i) \geq \frac{h^{'}N}{2} \text{ or } \sum_{i=1}^{n} p(i) \leq -\frac{h^{'}}{2(h+h^{'})}N \tag{12}$$

The proof follows using Lemma 8.4. □

Lemma
8.6. *Fix some constants $0 < h, h^{'} < 1$ such that $h + h^{'} < 1$. Fix some interval $Int = I_{j+1}, ..., I_{j+t}$ for some j, t, where $t \geq 1$. Assume that interval Int contains at least a fraction $(1-h)$ of all N projected points, where $N > \frac{4}{\lambda h^{'}}$. Then with probability at most $\nu(\frac{h^{'}N}{2(1-h-h^{'})}) + \nu(\frac{h^{'}N}{2})$ we have:*

$$\sum_{i=1}^{t}(m(j+i)+p(j+i)) \leq (1-h-h^{'})\sum_{i=1}^{n}(m(i)+p(i)),$$

where each $p(i) \sim L(0, 1/\lambda)$.

The proof of this lemma is similar to Lemma 8.5 and therefore is not provided.

Lemma
8.7. *Let $A \subset \mathbb{R}^D$ is contained in the ball $B(x_0, \Delta)$. Let $|A| = N$, where $N > \frac{40}{3\lambda}$. Then with probability at least $1 - (\frac{1}{20} + 2n(\nu(\frac{3N}{20}) + \nu(\frac{3N}{10})))$ the perturbed median M is within distance $\frac{(3.1+2\delta)\Delta}{\sqrt{D}}$ from $\hat{x_0}$ where $\delta = L/n$.*

Proof. By Lemma 8.2 we can say that with probability at least $(1 - \frac{1}{20})$, all but at most a $\frac{1}{5}$-fraction of all the projected data points are within an interval Int of center $\hat{x_0}$ and radius $\frac{3.1\Delta}{\sqrt{D}}$. Let $I_j, \ldots, I_{j+k}$ be the smallest sequence of interval segments that contain Int. Let a and b be the left and the right-ends of $I_j, \ldots, I_{j+k}$. Let E be the event that M is not within distance $\frac{(3.1+2\delta)\Delta}{\sqrt{D}}$ from $\hat{x_0}$. If E holds then either $M \leq a$ or $M \geq b$. In both cases, there exists some interval $I = I_1, I_2, ..., I_k$ or $I = I_k, I_{k+1}, ..., I_n$ for some $k \in \{1, 2, ...n\}$ such that $I \cap Int = \Phi$ and $\sum_{s:I_s \subseteq I}(m(s)+p(s)) \geq \frac{1}{2}\sum_{i=1}^{n}(m(s)+p(s))$. This holds with probability at most $\nu(\frac{3N}{20}) + \nu(\frac{3N}{10})$ which follows from Lemma 8.5 by choosing $h = \frac{1}{5}$ and $h^{'} = \frac{3}{10}$, where $N > \frac{4}{\lambda h^{'}}$. Since there are at most $2n$ intervals of the form I, the proof follows using union bound. □

Lemma
8.8. *Assume that the interval $I = I_{j+1}, ..., I_{j+t}$ contains all but at most a fraction $\frac{g}{2}$ of all N data points for some constant $g < \frac{1}{2}$. Then with probability at least $1 - 2n(\nu((\frac{1}{2}-g)N) + \nu(\frac{(\frac{1}{2}-g)N}{2}))$ the perturbed median M is within $I^* = I_j I_{j+1}, ..., I_{j+t}$.*

Proof. If $M \notin I^*$, then there exists interval $I' = I_1, ...I_k$ or $I' = I_k, ...I_n$, disjoint with I, such that $\sum_{s:I_s \subseteq I'}(m(s)+p(s)) \geq \frac{1}{2}\sum_{s=1}^{n}(m(s)+p(s))$ with probability at most $n((\frac{1}{2}-g)N) + n(\frac{(\frac{1}{2}-g)N}{2})$ (Lemma 8.6). Since there are at most $2n$ intervals of the form I', the proof follows using union bound. □

Properties of the Split

Given two balls B_i, B_j we say that a split is *good* if it completely separates them. A split is *bad* if the split point intersects both the balls. The remaining splits are called *neutral*.

The following lemma says that a split point chosen according to the rule *ChooseRule* in Algorithm 2 is of good quality. We first introduce the following notation: $\zeta_1 = \frac{20n}{3\lambda}$, $\zeta_2 = \frac{8n}{f\lambda}$, $\zeta_3 = \frac{4n}{(1-f)\lambda}$, $\zeta_4 = \frac{16}{\lambda f}$, $\zeta_5 = \psi_\lambda^n(\frac{5}{3ne^{15}})$, $\zeta_6 = \psi_\lambda^n(\frac{8}{fn^2e^{31}})$, $\zeta_7 = \psi_\lambda^n(\frac{8}{fe^{15}})$, $\zeta_8 = \psi_\lambda^n(\frac{1}{(1-f)ne^{15}}))$, $n = L/\delta$ with $\delta \leq \frac{0.1\Delta}{\sqrt{D}}$. Let $C = \sqrt{V \cdot K}$, where $V = 2(\sqrt{2\log(\frac{e^{15}}{g})} + 2\delta)$ and $g = \frac{1}{2}f$ is a constant as described in Algorithm 2.

Lemma

8.9. *Let $N > \max(\zeta_1, \zeta_2, \zeta_3, \zeta_4, \zeta_5, \zeta_6, \zeta_7, \zeta_8)$. Let $X \subseteq B(x_0, \Delta)$ have doubling dimension $d \geq 1$, and let X be (T, ρ, f)-good for $\rho > 65e^2$ and $T = (1 + 2e^2/\sqrt{d})W$. Assume that $\frac{0.00094e^{31}}{18V} > K > \frac{T}{\frac{1}{e^2} - \frac{65}{\rho}}$. Pick any two balls $B = B(z, r)$ and $B' = B(z', r)$ such that (i) their centers z and z' lie in $B(x_0, \Delta)$, (ii) the distance between these centers is at least $\frac{1}{2}\Delta - r$ and (iii) the radius r is at most $\frac{\Delta}{512C\sqrt{d}}$. Choose a split point according to the rule ChooseRule in Algorithm 2. Let p_l denote the probability that $X \bigcap B$ and $X \bigcap B'$ will completely be contained in separate halves of the split and p_u be the probability that the split point intersects both $X \bigcap B$ and $X \bigcap B'$. Then $p_d = p_l - 2p_u \geq \frac{0.00094}{VK} - \frac{18}{e^{31}} > 0$. The probabilities are taken over the choice of random directions U.*

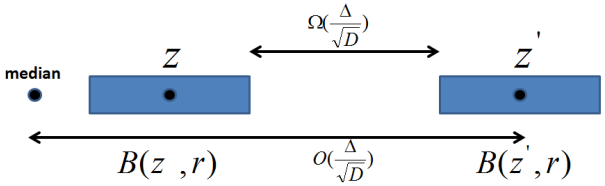

Figure 2: The setting as in Lemma 8.9. The key observation is that the projection images of the balls $B(z, r)$ and $B(z', r)$ are well-separable and their distance is at least of order $\Omega(\frac{\Delta}{\sqrt{D}})$. Thus, since the splitting point is chosen according to the random procedure, the probability that both projections with end up being on the different sides on the splitting point (which is proportional to the distance between the images) is lower-bounded by a positive constant.

Proof. Let U denote the random direction chosen in Algorithm 2 (in Algorithm 2, we choose 15 random directions: 14 of them to compute the median and the fifteenth one to choose the split point). Choose $\delta_1 = \frac{2}{e^{31}}$ and $\delta_2 = \frac{1}{20}$. Denote by $\hat{B}$ and $\hat{B}'$ the projections of $X \bigcap B$ and $X \bigcap B'$ respectively on U. Notice that:

1. With probability at least $1 - 2\delta_1$, both $\hat{B}$ and $\hat{B}'$ are contained within intervals of radius at most $\frac{\Delta}{16C\sqrt{D}}$ around $\hat{z}$ and $\hat{z}'$ respectively (consequence of Lemma 8.2).
2. With probability at least 3/5, $|\hat{z} - \hat{z'}| \geq \frac{\Delta}{4\sqrt{D}}$ (consequence of Lemma 8.1).
3. With probability at least $1 - p_3$, where $p_3 = \frac{2\delta_2}{\sqrt{2\log(\frac{2}{\delta_2})}}$ both $\hat{z}$ and $\hat{z}'$ are within distance $\frac{3\Delta}{\sqrt{D}}$ of $\hat{x_0}$, where $\hat{x_0}$ is the projection of x_0 (consequence of Lemma 8.1).
4. The distance of the perturbed median of $\hat{X}$ from $\hat{x_0}$ is at most $\frac{3.3\Delta}{\sqrt{D}}$ with probability at least $p_4 \leq (\frac{1}{20} + 2n(\nu(\frac{3N}{20}) + \nu(\frac{3N}{10})) + \nu(N))$ (consequence of Lemma 8.7).

We can now use the above properties to prove $p_l - 2p_u \geq 0$. Recall that the procedure **ChooseRadius** in Algorithm 2 computes an interval with the perturbed median as its center that is g-good. Let E_1 be the event that radius of the interval is at most $\frac{\Delta}{K\sqrt{D}}$ and E_2 be the event that the radius is larger than $V\frac{\Delta}{\sqrt{D}}$.

Let $\theta_1 = P(E_1)$ and $\theta_2 = P(E_2)$. We say that a random direction U is *good* if properties 1,2,3 and 4 are satisfied and E_1 and E_2 do not hold. Let $p_g = P(U \text{ is good})$ and $p_c = P(B, B^{'} \text{ are cleanly separated } |U \text{ is good})$. Then $p_l \geq p_g p_c$. It can be easily shown that

$$p_l \geq (1 - \frac{2}{5} - \delta_2 - \frac{2\delta_2}{\sqrt{2\log(\frac{2}{\delta_2})}} + p^e)r^e, \tag{13}$$

where:

$$p^e = -2\delta_1 - 2n(\nu(\frac{3N}{20}) + \nu(\frac{3N}{10})) - \nu(N) - \theta_1 - \theta_2 \tag{14}$$

and

$$r^e = \frac{\frac{\Delta}{4\sqrt{D}} - \frac{2\Delta}{16C\sqrt{D}}}{\frac{2\cdot 6.6K\Delta\cdot V}{\sqrt{D}}} \tag{15}$$

Using the observations $P[|\hat{z} - \hat{z}^{'}| \leq \frac{\Delta}{8C\sqrt{D}}] \leq \sqrt{\frac{2}{\Pi}}\frac{64}{255C}$ and $P[\text{split point intersects } \hat{B}] \leq \frac{1}{6.6\cdot 16\cdot C}$, we can show that

$$p_u \leq (\sqrt{\frac{2}{\Pi}}\frac{64}{255C})(\frac{1}{6.6\cdot 16\cdot C}) + 2\delta_1 + \theta_1 \tag{16}$$

We can prove that $\theta_1 \leq \frac{5}{e^{31}}$ and $\theta_2 \leq \frac{70}{e^{15}}$ and the proofs are provided below. Thus it is true that $p_d = p_l - 2p_u \geq \frac{0.00094}{VK} - \frac{18}{e^{31}} > 0$ for $K < \frac{0.00094e^{31}}{18V}$. □

Proof of the Claim $\theta_1 \leq \frac{5}{e^{31}}$.

Proof. Choose a random direction U. (Recall that this is one of the random projections chosen in the procedure **ChooseRule** of Algorithm 2. Since X is (T, ρ, f)-good, by definition, there exist two balls in X, namely: $C_1(c_1, l)$ and $C_1(c_2, l)$ each of which contain at least fN points of X and is such that $di = dist(c_1, c_2) \geq \frac{\Delta}{T}$ and $l \leq \frac{di}{\rho\sqrt{d}}$. Choosing $\delta_3 = \frac{1}{e^{31}}$ and using Lemma 8.3, we can conclude that with probability at least $1 - \delta_3$ the projections of C_1 and C_2 lies within segments of radius $\frac{4di}{\rho\sqrt{D}}\sqrt{2\log(\frac{2e}{\delta_3})}$.

If I^e is a segment centered at M and of radius $\hat{R}$ and does not intersect the projection of C_1 or the projection of C_2 then Int contains at most $(1-f)N$ projected data points. This follows from the fact that X is (T, ρ, f)-good. Let I^e denote any fixed interval containing at most $(1-f)N$ points. Choosing $h = (1-f)$, $h^{'} = \frac{f}{2}$, and using the assumption $N > \frac{16}{\lambda f}$ it follows from Lemma 8.5 that with probability at most $\nu(\frac{fN}{4}) + \nu(\frac{fN}{2(2-f)})$ the following holds: $\sum_{i:I_i \subseteq I^e}(m(i) + p(i)) \geq (1-g)\sum_{i=1}^{n}(m(i) + p(i))$. There are at most n^2 possible intervals I^e formed by some number (possibly 0) of consecutive subintervals I_i. Hence, with probability at least $1 - (\sqrt{\frac{2}{\Pi}}\alpha + 2\delta_3) - (\nu(\frac{fN}{4}) + \nu(\frac{fN}{2(2-f)}))n^2$ every point from the projection of C_1 is at distance at least $\frac{\Delta}{K\sqrt{D}}$ from every point from the projection of C_2 and the interval I^e intersects C_1 and C_2. This implies $\hat{R} \geq \frac{\Delta}{K\sqrt{D}}$ with probability at least $1 - (\sqrt{\frac{2}{\Pi}}\alpha + 2\delta_3) - (\nu(\frac{fN}{4}) + \nu(\frac{fN}{2(2-f)}))n^2$. Since the *rad* is calculated as the maximum over all $\hat{R}$ on 14 independently chosen random directions U, $\theta_1 \leq ((n(\frac{fN}{4}) + n(\frac{fN}{2(2-f)}))n^2 + (\sqrt{\frac{2}{\Pi}}\alpha + 2\delta_3))^{14}$. This implies $\theta_1 \leq \frac{5}{e^{31}}$ from our assumptions on N. □

Proof of the Claim $\theta_2 \leq \frac{70}{e^{15}}$.

Proof. Fix i^{th} random direction U. From Lemma 8.2, we know that with probability at least $1 - \frac{2}{e^{15}}$, all but at most a h-fraction of all N points are within distance at most $d_1 = \frac{\Delta}{\sqrt{D}}\sqrt{2\log(\frac{e^{15}}{2h})}$ from $\hat{x_0}$. Let Int denote the interval centred at $\hat{x_0}$ and radius d_1. Let Int^u denote the shortest interval centred at $\hat{x_0}$, and radius at least d_1 that contains Int such that Int^u is the union of sub-segments I_i. The radius of Int^u is at most $d_1 + \delta$.

Choosing $h = h^{'} = \frac{g}{2}$, and assuming $N > \frac{16}{\lambda f}$, using Lemma 8.6 and 8.8, we can show that the probability for any U chosen from the set of 14 random directions, that radius $\hat{R}$ is larger than $V\frac{\Delta}{\sqrt{D}}$ is at most $p_m = (\frac{2}{e^{15}} + \nu(\frac{gN}{4(1-g)}) + \nu(\frac{gN}{4}) + 2(\nu(\frac{(\frac{1}{2}-g)N}{2}) + \nu(\frac{1}{2} - g)N)n)$. So it suffices to show that $p_m \leq \frac{5}{e^{15}}$. But that follows directly from our assumption on N and hence $\theta_2 \leq \frac{70}{e^{15}}$. □

Proof of Theorem 7.2: Cover $X \cap A$ by balls of radius $r = \Delta/(512C\sqrt{d})$, where C is a constant defined in Lemma 8.9. Since $X \cap A$ has doubling dimension d, $X \cap A$ is covered by at most $N_b = (O(d))^d$ balls. Fix any pair of balls B, $B^{'}$ from this cover whose centers are at distance at least $\frac{\Delta}{2} - r$ from one another. Let p_k be the probability that there exists some cell k levels below A which contains points from both B and $B^{'}$ $(k = 1, 2, ...)$. Let p_l and p_u be the probabilities defined in Lemma 8.9. To apply Lemma 8.9, first we need to prove that if $X \cap A$ is not (η, W)-dense then $X \cap A$ is (T, ρ, f)-good. We do it the following way: cover $X \cap A$ by CM^d balls, each of radius $\frac{\Delta}{M}$ where $M = 130e^2T\sqrt{d}$. There exist at least one ball that contains at least $\frac{N}{CM^d}$ points. Denote this ball by $B_1(x_0, \frac{\Delta}{M})$. Consider all balls with centers outside B_2. If those balls together contain at most ηN points then at least $(1 - \eta)N$ points are within ball $B_3(x_0, \frac{\Delta}{T} + \frac{\Delta}{M})$ implying that $X \cap A$ is (η, W)-dense, which is a contradiction. So the balls with centers outside B_2 contain altogether at least ηN points. One of them, denote it by B_4, contains at least $\frac{\eta}{CM^d}N$ points. Let $f = \frac{\eta}{CM^d}$, $\rho = \frac{M}{T\sqrt{d}}$. Using balls B_1 and B_4 we can conclude that $X \cap A$ is (T, ρ, f)-good. Now, we are ready to apply Lemma 8.9. It follows from Lemma 8.9 that for $k > 1$: $p_k \leq p_l \cdot 0 + p_u \cdot 2p_{k-1} + (1 - p_l) \cdot p_{k-1}$ and $p_k \leq wp_{k-1}$, where $0 < w = (1 - (p_u - 2p_l)) < 1$. Thus for some constant $c^{'}$ and $k = c^{'}d\log(d)$, we have $p_k \leq \frac{1}{N_b^2}$. Taking the union bound over all pairs of balls from the cover which are at the prescribed minimum distance from each other completes the proof. ∎

References

[1] Adam, N. R. and Worthmann, J. C. (1989). Security-control methods for statistical databases: A comparative study. *ACM Comput. Surv.*, 21(4):515–556.

[2] Agrawal, R. and Srikant, R. (2000). Privacy-preserving data mining. In *SIGMOID '00*, volume 29, pages 439–450.

[3] Beimel, A., Nissim, K., and Stemmer, U. (2013). Private learning and sanitization: Pure vs. approximate differential privacy. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 16th International Workshop, APPROX 2013, and 17th International Workshop, RANDOM 2013, Berkeley, CA, USA, August 21-23, 2013. Proceedings*, pages 363–378.

[4] Bishop, C. (2006). *Pattern Recognition and Machine learning*. Springer.

[5] Blum, A., Dwork, C., McSherry, F., and Nissim, K. (2005). Practical privacy: The SuLQ framework. In *PODS '05*, pages 128–138.

[6] Blum, A., Ligett, K., and Roth, A. (2008). A learning theory approach to non-interactive database privacy. In *STOC '08*, pages 609–618.

[7] Brickell, J. and Shmatikov, V. (2008). The cost of privacy: Destruction of data-mining utility in anonymized data publishing. In *KDD '08*, pages 70–78.

[8] Chaudhuri, K. and Hsu, D. (2011). Sample complexity bounds for differentially private learning. *Journal of Machine Learning Research*, 19:155–186.

[9] Chaudhuri, K., Monteleoni, C., and Sarwate, A. (2011). Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12:1069–1109.

[10] Chaudhuri, K., Sarwate, A. D., and Sinha, K. (2012). Near-optimal algorithms for differentially-private principal components. *CoRR*, abs/1207.2812.

[11] Cormode, G., Procopiuc, C. M., Srivastava, D., and Tran, T. T. L. (2011). Differentially private publication of sparse data. *CoRR*, abs/1103.0825.

[12] Cormode, G., Procopiuc, M., Shen, E., Srivastava, D., and Yu, T. (2012). Differentially private spatial decompositions. In *ICDE*, pages 20–31.

[13] Cox, T. and Cox, M. (2000). *Multidimensional Scaling*. Chapman and Hall.

[14] Dasgupta, S. and Freund, Y. (2008). Random projection trees and low dimensional manifolds. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, STOC '08, pages 537–546.
[15] Dwork, C. (2008). Differential privacy: A survey of results. In *TAMC : Theory and Applications of Models of Computation, 5th International Conference*, pages 1–19.
[16] Dwork, C. (2009). The differential privacy frontier (extended abstract). In *TCC*, pages 496–502.
[17] Dwork, C. (2010). Differential privacy in new settings. In *SODA*, pages 174–183.
[18] Dwork, C. (2011). A firm foundation for private data analysis. *Commun. ACM*, 54(1):86–95.
[19] Dwork, C. and Lei, J. (2009). Differential privacy and robust statistics. In *STOC '09*, pages 371–380.
[20] Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284.
[21] Feldman, D., Fiat, A., Kaplan, H., and Nissim, K. (2009a). Private coresets. In *STOC '09*, pages 361–370.
[22] Feldman, D., Fiat, A., Kaplan, H., and Nissim, K. (2009b). Private coresets. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 361–370.
[23] Freund, Y., Dasgupta, S., Kabra, M., and Verma, N. (2007). Learning the structure of manifolds using random projections. In *NIPS*.
[24] Friedman, A. and Schuster, A. (2010). Data mining with differential privacy. In *KDD*, pages 493–502.
[25] Ganta, S., Kasiviswanathan, S., and Smith, A. (2008). Composition attacks and auxiliary information in data privacy. In *KDD '08*.
[26] Goldreich, O. (2004). *Foundations of Cryptography, Vol II*. Cambridge University Press.
[27] Gupta, A., Ligett, K., McSherry, F., Roth, A., and Talwar, K. (2010). Differentially private combinatorial optimization. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1106–1125.
[28] Inan, A., Kantarcioglu, M., Ghinita, G., and Bertino, E. (2010). Private record matching using differential privacy. *(EDBT*.
[29] Jagannathan, G., Pillaipakkamnatt, K., and Wright, R. N. (2009). A practical differentially private random decision tree classifier. In *ICDMW '09: Proceedings of the 2009 ICDM Workshops*, pages 114–121.
[30] Jain, P., Kothari, P., and Thakurta, A. (2012). Differentially private online learning. *COLT*.
[31] Jain, P. and Thakurta, A. (2013). Differentially private learning with kernels. *ICML-to appear*.
[32] Kapralov, M. and Talwar, K. (2013). On differentially private low rank approximations. *SODA*.
[33] Kasiviswanathan, S., Lee, H. K., Nissim, K., Raskhodnikova, S., and Smith, A. (2008). What can we learn privately? In *FOCS '08: Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 531–540.
[34] Kpotufe, S. (2009). Escaping the curse of dimensionality with a tree-based regressor. In *Conference on Computational Learning Theory*.
[35] Lee, J. A. and Verleysen, M. (2007). *Nonlinear Dimensionality Reduction*. Springer.
[36] Lindell, Y. and Pinkas, B. (2002). Privacy preserving data mining. *J. Cryptology*, 15(3):177–206.
[37] McSherry, F. and Talwar, K. (2007). Mechanism design via differential privacy. In *FOCS '07: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 94–103.
[38] Nissim, K., Raskhodnikova, S., and Smith, A. (2007). Smooth sensitivity and sampling in private data analysis. In *STOC '07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 75–84, New York, NY, USA. ACM.
[39] Peng, S., Yang, Y., Zhang, Z., Winslett, M., and Yu, Y. (2012). Dp-tree: indexing multi-dimensional data under differential privacy (abstract only). In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*, page 864.
[40] Samarati, P. and Sweeney, L. (1998). Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical Report SRI-CSL-98-04, SRI Computer Science Laboratory.
[41] Sweeney, L. (2002). k-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570.