# Differentially-Private Learning of Low Dimensional Manifolds

Anna Choromanska[1], Krzysztof Choromanski[2], Geetha Jagannathan[3], and 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, DC, USA
{aec2163,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 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.

## 1 Introduction

Many real world datasets are measured at extremely high dimension. Analyzing datasets in 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 (2006); Cox and Cox (2000)). 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 (2007)) provide ways to construct mappings from the given high dimensional spaces into the low dimensional manifolds in which the data actually lie.

S. Jain et al. (Eds.): ALT 2013, LNAI 8139, pp. 249–263, 2013.

Dasgupta and Freund (2008) analyzed the technique presented by Freund et al. (2007), 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 (2009) 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. (2006) 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 differentially-private learning. Differentially private data analysis needs more data than its non-private counterpart to achieve a comparable amount of accuracy. The amount of data required in high dimensional space for a differentially private learning becomes exorbitant.

## 2 Our Contribution

In this paper, we focus on data of low doubling dimension as was considered by Dasgupta and Freund (2008). We give a differentially-private manifold learning algorithm that constructs a differentially-private data structure that depends only on the doubling dimension of the data. 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 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 which 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 (2008). Our algorithm, as well as the algorithm presented by Dasgupta and Freund (2008), 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 (1989); Agrawal and Srikant (2000)), $k$-anonymity and its variants (Samarati and Sweeney (1998); Sweeney (2002)) and secure multiparty computation (Goldreich (2004); Lindell and Pinkas (2002)). The weakness of these privacy models have also been well studied (Ganta et al. (2008); Brickell and Shmatikov (2008)). The differential privacy framework introduced by Dwork et al. (2006) 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. (2006) significant amount of work has been done in this area and most of them are surveyed by Dwork (2008, 2009, 2010, 2011).

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. (2005)), which is a predecessor of differential privacy. Ignoring computational constraints, Kasiviswanathan et al. (2008) showed that anything which is PAC-learnable is also differentially-private PAC-learnable. Building upon their technique, Blum et al. (2008) 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. (2011) showed that it is possible to obtain differentially-private empirical risk minimization algorithms by perturbing their objective functions. Feldman et al. (2009) 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. (2009) presented a differentially private classifier based on random decision trees. Their algorithm achieves good accuracy even for small datasets. Friedman and Schuster (2010) presented a differentially private ID3 algorithm that gives better accuracies than the straightforward construction of a differentially private ID3 tree. Recently, Chaudhuri and Hsu (2011) 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. However there are several papers where differentially-private constructions of other important structures are presented. Cormode et al. (2011) consider the problem of differentially private release of sparse data. Chaudhuri et al. (2012) investigated the performance of differentially private principal component analysis which is used for dimensionality reduction. Finally, very recently Kapralov and Talwar (2013)

presented an algorithm that outputs a differentially-private approximation to the principal eigenvector of a given symmetric matrix. Differential privacy has been also studied in the online learning context (see for example: Jain et al. (2012)). Another interesting setting involves scenario, where the access to the training features is only through a kernel function (see: Jain and Thakurta (2013)).

## 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 1 (Dwork et al. (2006)).** *A randomized algorithm $\mathcal{M}$ satisfies* $\epsilon$-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 $\epsilon$ 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 *$\epsilon$-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 2 (Dwork et al. (2006)).** *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 $\lambda$.

**Theorem 1 (Dwork et al. (2006)).** *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 $\epsilon$-differential privacy.*

Using this method, smaller values of $\epsilon$ 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 2 ((Dwork et al., 2006)).** ***(Composition Theorem)*** *The sequential application of mechanisms $\mathcal{M}_i$, each giving $\epsilon_i$-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 $\ell$. We assume $\ell$ 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 (2008)) 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 (2008) 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 (2008) 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 (2008). 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 3.** *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 $r/2$.*

The following theorem is the main ingredient of the proof that random projection tree constructed in Dasgupta and Freund (2008) is of good quality. We will obtain similar result for a differentially-private random projection tree constructed by us in this paper.

**Theorem 3 (Dasgupta and Freund (2008)).** *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* $radius(C') < radius(C)$*.*

**Algorithm 1.** RP Tree Algorithm [Dasgupta and Freund (2008)]

**Algorithm MakeTree**
Input: X, the data set,
Output: A decision tree

```
if |X| < MinSize then
  return (Leaf)
else
  Rule ←ChooseRule(X)
  LeftTree ←MakeTree({x ∈ X : Rule(x) = true})
  RightTree ←MakeTree({x ∈ X : Rule(x) = false})
  return ([Rule, LeftTree, RightTree])
end if
```

***Subroutine*** **ChooseRule**$(X)$

```
choose a random unit direction v ∈ R^D
pick any point x ∈ X, and let y be the farthest point from it in X
choose s uniformly at random in [−1, 1] · 6||x − y||/√D
Rule(x) := x · v ≤ (median({z · v : z ∈ X}) + s)
return (Rule)
```

## 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 (2008). 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 4.** *A set* $X$ *of* $N$ *points and diameter* $\Delta$ *is* $(\eta, W)$*-dense if at least* $(1-\eta)N$ *points of* $X$ *are within a ball of radius* $\frac{\Delta}{W}$*.*

**Definition 5.** *Consider a sequence of real numbers* $(\beta_1, ..., \beta_n)$ *where each* $\beta_i$ *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.** *A set* $X$ *of diameter* $\Delta$ *and doubling dimension* $d$ *is* $(T, \rho, f)$*-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* $X$*.*

**Algorithm 2.** Differentially Private RP Tree Algorithm (**DP-RPtree**)

**Algorithm MakeDPRPTree**
Input: $X$, the data set, $h$, the height of the current node, constants $K$ and $g < 1/2$
Output: A decision tree

**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 if**

***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** $i \in \{1, \ldots, 14\}$ **do**
  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**
    $radius \leftarrow R$
  **end if**
**end for**
**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 $\sum_{i=1}^{j}(m_i + p_i) \geq \frac{1}{2}N$
**end if**
**return** left end point of $I_j$

Before describing our algorithm, we will first explain a naive conversion of the RP tree algorithm of Dasgupta and Freund (2008) 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 2, one can show that such a construction is differentially-private. The following computations in the procedure **ChooseRule**$(X)$ in Algorithm 1 require 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.

One can replace both the diameter and the median computations with differentially private diameter and median computations (Dwork et al. (2006)). The direct way of doing this is by computing the median and the diameter of the input dataset $S$, and then adding an appropriate noise term chosen from a Laplace distribution, using a parameter that is linearly dependent on the global sensitivity of these queries. However, both median and diameter have high global sensitivity. This substantially impacts the precision of the differentially-private median and diameter computations. The trees built from them can be correspondingly poor. One could use the exponential mechanism of McSherry and Talwar (2007) to compute differentially-private median. However, the exponential mechanism provides a good split with constant probability only if the data is not skewed (Cormode et al. (2012)). The differentially private median computed using smooth sensitivity (Nissim et al. (2007)) offers a weaker privacy guarantee $(\epsilon, \delta)$-differential privacy than the standard differential privacy model. The Propose-Test-Release mechanism proposed by Dwork and Lei (2009) also provides $(\epsilon, \delta)$-differential privacy. Further, 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. Hence, in this paper we present a procedure to compute approximations to the median using only low sensitivity count queries. We use a much weaker assumption on the density of the data, 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 our construction, we avoid these 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 $\ell$, where $\ell$ 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 $\ell$ 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 $\delta$ (we have: $\delta = \frac{L}{n}$). We call $\delta$ a *precision parameter*. Both $n$ and $\delta$ are public. We compute differentially private counts of the projected points $\beta_i$ 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, $\beta_i$ 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.

**Theorem 4. DP-RPtree** *is* $14h\lambda$ *differentially-private where* $h$ *is the height of the tree and* $\lambda$ *is the differentially-private parameter.*

*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 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 5) and also a slightly stronger result (Theorem 6) 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 $\rho_A$ 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 $C$.

**Theorem 5.** *Let $X \subseteq R^D$ and $\lambda$ denote the differential-privacy parameter. Pick any cell $A$ of the differentially-private RP tree. Suppose $X \cap A$ has doubling dimension $\leq d$, has diameter $2\Delta$ and contains $N$ points. Assume that 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 and $N = \Omega(\frac{e^{2d} d^{\frac{d}{2}} n \log^2(n)}{\lambda})$ for $n = \frac{L}{\delta}$. Assume furthermore that algorithm parameters $g$ and $\delta$ are small enough and $K = 400W$. 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 (2008) while preserving differential privacy. Note that the smaller the precision parameter $\delta$, the bigger $n$ and thus we need more data points in the theorem. This agrees with our inuition since smaller length $\delta$ 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 5. This result is stated in Theorem 6 that we now provide. Let $\nu(x) = \frac{(\lambda x)^n}{\exp(\lambda x)}$ and $\psi_\lambda^n$ is the inverse of $\nu(x)$, defined on $[\frac{n}{\lambda}, \infty]$.

**Theorem 6.** *Let $X \subseteq R^D$ and $\lambda$ denote the differential-privacy parameter. Pick any cell $A$ of the differentially-private RP tree. Suppose $X \cap A$ has doubling dimension $\leq d$, has diameter $2\Delta$ and contains $N$ points. Assume that the set $X \cap A$ is not $(\eta, W)$-dense for some constant $W$, where $0 \leq \eta \leq 1$ and*

$$N > \max(\zeta_1, \zeta_2, \zeta_3, \zeta_4, \zeta_5, \zeta_6, \zeta_7, \zeta_8),$$

*where $\zeta_1 = \frac{20n}{3\lambda}$, $\zeta_2 = \frac{8n}{f\lambda}$, $\zeta_3 = \frac{4n}{(1-f)\lambda}$, $\zeta_4 = \frac{16}{\lambda f}$, $\zeta_5 = \psi^n_\lambda(\frac{5}{3ne^{15}})$, $\zeta_6 = \psi^n_\lambda(\frac{8}{fn^2e^{31}})$, $\zeta_7 = \psi^n_\lambda(\frac{8}{fe^{15}})$, $\zeta_8 = \psi^n_\lambda(\frac{1}{(1-f)ne^{15}}))$, $n = L/\delta$ and $f = \frac{\eta}{CM^d}$ for $M = 130e^2\sqrt{d}$ and some constant $C$. Denote $T = (1+\frac{2e^2}{\sqrt{d}})W$. Assume that constants $\delta, g, K$ from the Algorithm 2 satisfy: $g < \frac{\eta}{2C}(\frac{1}{130e^2T\sqrt{d}})^d$, $\zeta_9 > K > 2e^2T$ for $\zeta_9 = \frac{0.00094e^{31}}{18(\sqrt{30+2\log(\frac{2C}{\eta})+2d\log(130e^2T\sqrt{d})+2\delta})}$ and $\delta \leq \frac{0.1\Delta}{\sqrt{D}}$. 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 6 implies Theorem 5 as follows: If $X \cap A$ is $(\eta, W)$-dense then there exists a ball $B$ of radius $\frac{\Delta}{W}$ that contains all but at most $\eta N$ 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 5.

We give here a brief outline of the proof of Theorem 6 (the formal proof can be found in the extended version of the paper). 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 (2008), 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 (2008), 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.

In the Appendix we present some properties of the perturbed median and of the split, used in the proof of the main result and the proof of the main result. Missing proofs of the technical results may be found in the extended version of the paper.

# 8 Appendix

**Properties of the Perturbed Median**

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.

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.
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 3).
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 4).

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

**Lemma 1.** *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: $P[l_1 + ... + l_n \geq W] \leq \frac{(\lambda W)^n}{e^{\lambda W}}$.*

**Lemma 2.** *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 $p(i) \sim L(0, 1/\lambda)$.*

**Lemma 3.** *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.* It can be proven 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 $\nu(\frac{3N}{20}) + \nu(\frac{3N}{10})$ 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 4.** *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 $I^* = I_j I_{j+1}, ..., I_{j+t}$.*

### 8.1 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*.

**Lemma 5.** *Let $X \subseteq B(x_0, \Delta)$ have doubling dimension $d \geq 1$. Let $X$ be $(T, \rho, f)$-good for $\rho > 65e^2$, $T = (1 + 2e^2/\sqrt{d})W$ and*

$$N > \max(\zeta_1, \zeta_2, \zeta_3, \zeta_4, \zeta_5, \zeta_6, \zeta_7, \zeta_8)$$

*, where $\zeta_1 = \frac{20n}{3\lambda}$, $\zeta_2 = \frac{8n}{f\lambda}$, $\zeta_3 = \frac{4n}{(1-f)\lambda}$, $\zeta_4 = \frac{16}{\lambda f}$, $\zeta_5 = \psi_\lambda^n(\frac{5}{3ne^{15}})$, $\zeta_6 = \psi_\lambda^n(\frac{8}{fn^2e^{31}})$, $\zeta_7 = \psi_\lambda^n(\frac{8}{fe^{15}})$, $\zeta_8 = \psi_\lambda^n(\frac{1}{(1-f)ne^{15}})$), $n = L/\delta$ with $\delta \leq \frac{0.1\Delta}{\sqrt{D}}$. Assume that $\frac{0.00094e^{31}}{18V} > K > \frac{T}{\frac{1}{e^2} - \frac{65}{\rho}}$. 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. 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 $U$.*

### 8.2 Proof of the Main Theorem

**Proof of Theorem 6:** Cover $X \cap A$ by balls of radius $r = \Delta/(512C\sqrt{d})$, where $C$ is a constant defined in Lemma 5. 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 5. To apply Lemma 5, first we need to prove that if $X \cap A$ is not $(\eta, W)$-dense then $X \cap A$ is $(T, \rho, f)$-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 $\eta N$ 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 $(\eta, W)$-dense, which is a contradiction. So the balls with centers outside $B_2$ contain altogether at least $\eta N$ 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 $(T, \rho, f)$-good. Now, we are ready to apply Lemma 5. It follows from Lemma 5 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


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

Agrawal, R., Srikant, R.: Privacy-preserving data mining. In: SIGMOID 2000, vol. 29, pp. 439–450 (2000)

Bishop, C.: Pattern Recognition and Machine learning. Springer (2006)

Blum, A., Dwork, C., McSherry, F., Nissim, K.: Practical privacy: The SuLQ framework. In: PODS 2005, pp. 128–138 (2005)

Blum, A., Ligett, K., Roth, A.: A learning theory approach to non-interactive database privacy. In: STOC 2008, pp. 609–618 (2008)

Brickell, J., Shmatikov, V.: The cost of privacy: Destruction of data-mining utility in anonymized data publishing. In: KDD 2008, pp. 70–78 (2008)

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

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

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

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

Cormode, G., Procopiuc, M., Shen, E., Srivastava, D., Yu, T.: Differentially private spatial decompositions. In: ICDE, pp. 20–31 (2012)

Cox, T., Cox, M.: Multidimensional Scaling. Chapman and Hall (2000)

Dasgupta, S., Freund, Y.: Random projection trees and low dimensional manifolds. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC 2008, pp. 537–546 (2008)

Dwork, C.: Differential privacy: A survey of results. In: 5th International Conference on TAMC: Theory and Applications of Models of Computation, pp. 1–19 (2008)

Dwork, C.: The differential privacy frontier (extended abstract). In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 496–502. Springer, Heidelberg (2009)

Dwork, C.: Differential privacy in new settings. In: SODA, pp. 174–183 (2010)

Dwork, C.: A firm foundation for private data analysis. Commun. ACM 54(1), 86–95 (2011)

Dwork, C., Lei, J.: Differential privacy and robust statistics. In: STOC 2009, pp. 371–380 (2009)

Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)

Feldman, D., Fiat, A., Kaplan, H., Nissim, K.: Private coresets. In: STOC 2009, pp. 361–370 (2009)

Freund, Y., Dasgupta, S., Kabra, M., Verma, N.: Learning the structure of manifolds using random projections. In: NIPS (2007)
Friedman, A., Schuster, A.: Data mining with differential privacy. In: KDD, pp. 493–502 (2010)
Ganta, S., Kasiviswanathan, S., Smith, A.: Composition attacks and auxiliary information in data privacy. In: KDD 2008 (2008)
Goldreich, O.: Foundations of Cryptography, vol. II. Cambridge University Press (2004)
Jagannathan, G., Pillaipakkamnatt, K., Wright, R.N.: A practical differentially private random decision tree classifier. In: ICDMW 2009: Proceedings of the 2009 ICDM Workshops, pp. 114–121 (2009)
Jain, P., Kothari, P., Thakurta, A.: Differentially private online learning. Journal of Machine Learning Research - Proceedings Track 23, 24.1–24.34 (2012)
Jain, P., Thakurta, A.: Differentially private learning with kernels. ICML (to appear, 2013)
Kapralov, M., Talwar, K.: On differentially private low rank approximation. In: SODA, pp. 1395–1414 (2013)
Kasiviswanathan, S., Lee, H.K., Nissim, K., Raskhodnikova, S., Smith, A.: What can we learn privately? In: FOCS 2008: Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science, pp. 531–540 (2008)
Kpotufe, S.: Escaping the curse of dimensionality with a tree-based regressor. In: COLT (2009)
Lee, J.A., Verleysen, M.: Nonlinear Dimensionality Reduction. Springer (2007)
Lindell, Y., Pinkas, B.: Privacy preserving data mining. J. Cryptology 15(3), 177–206 (2002)
McSherry, F., Talwar, K.: Mechanism design via differential privacy. In: FOCS 2007: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, pp. 94–103 (2007)
Nissim, K., Raskhodnikova, S., Smith, A.: Smooth sensitivity and sampling in private data analysis. In: STOC 2007: Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing, pp. 75–84. ACM, New York (2007)
Samarati, P., Sweeney, L.: Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical Report SRI-CSL-98-04, SRI Computer Science Laboratory (1998)
Sweeney, L.: $k$-anonymity: A model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 10(5), 557–570 (2002)