Private Covariance Estimation via Iterative Eigenvector Sampling

Kareem Amin
Google Research NY
kamin@google.com

Travis Dick
Carnegie Mellon University
tdick@cs.cmu.edu

Alex Kulesza
Google Research NY
kulesza@google.com

Andrés Muñoz Medina
Google Research NY
ammedina@google.com

Sergei Vassilvitskii
Google Research NY
sergeiv@google.com

Abstract

We give a new method for computing a differentially private covariance matrix, suitable for use in a variety of regression settings, from a collection of user data. In contrast to previous methods, our approach has zero failure probability, and can be employed to compute the full matrix (not just the top k eigenvectors). As part of our analysis, we derive a precise expression for allocating the privacy budget ϵ among different components to minimize overall error. We then show empirically that our method outperforms previous state-of-the-art approaches, yielding lower error and better learning performance.

1 Introduction

Differential privacy has emerged as a standard framework for thinking about user privacy in the context of large scale data analysis [Dwork et al., 2014a]. While differential privacy does not protect against all attack vectors, it does provide formal guarantees about possible information leakage. A key feature of differential privacy is its robustness to post-processing: once a dataset is certified as differentially private, arbitrary post-processing can be performed without additional privacy impact.

The past decade has seen the emergence of a wide range of techniques for modifying classical learning algorithms to be differentially private [McSherry and Mironov, 2009, Chaudhuri et al., 2011, Jain et al., 2012, Abadi et al., 2016]. These algorithms typically train directly on the raw data, but are guaranteed to produce differentially private outputs. An alternative approach is to first preprocess the dataset into a differentially private form and then freely choose among standard off-the-shelf algorithms for learning. This not only provides more flexibility in the design of the learning system, but also removes the need for access to sensitive raw data (except for the initial preprocessing step). Thus, this approach falls under the umbrella of "data release": since the preprocessed dataset is differentially private, it can (in principle) be released without leaking any individual's data.

In this work we consider the problem of computing, in a differentially private way, a specific representation of a dataset: its covariance matrix. Formally, given a data matrix $\mathbf{X} \in \mathbb{R}^{d \times n}$, each column corresponds to a data point, we aim to compute a private version of $\mathbf{C} = \mathbf{X}\mathbf{X}^\top$, which can be used in place of the raw data as the basis for standard linear regression algorithms. Our methods provide privacy guarantees for the columns of $\mathbf{X}$.

A naïve approach using a standard differential privacy mechanism would be to simply add an appropriate amount of Laplace noise independently to every element in the true covariance matrix $\mathbf{C}$. However, the amount of noise required makes such a mechanism impractical. A better approach is to add Gaussian noise [Dwork et al., 2014b]; however, this results in (ϵ, δ)-differential privacy,

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

where, with some probability δ, user privacy can be violated. Another approach, proposed in Chaudhuri et al. [2012], is to compute a private version of PCA. This approach has two major flaws when applied to our setting. First, it only works for computing the top eigenvectors, and fails to give non-trivial results for computing the full covariance matrix. Second, the sampling itself is quite involved and requires the use of a Gibbs sampler. Since it is generally impossible to know when the sampler converges, adding noise in this way can seriously compromise the privacy guarantees of the mechanism.

The approach we propose in this work guarantees pure ϵ-differential privacy, can be used to obtain a full-rank covariance matrix, is practical, and, empirically, leads to lower approximation errors than the two mentioned baselines. On the technical side, we show how the overall privacy budget of ϵ should be split among the d eigenvectors in order to reduce overall error.

Formal Problem Setting and Notation. Let $\mathbf{X} \in \mathbb{R}^{d \times n}$ be a data matrix where each column corresponds to a d-dimensional data point. Our goal is to privately release an estimate of the unnormalized and uncentered covariance matrix $\mathbf{C} = \mathbf{X}\mathbf{X}^\top \in \mathbb{R}^{d \times d}$. In particular, we assume that the columns of $\mathbf{X}$ have ℓ_2-norm bounded by 1, and we say that two data matrices $\mathbf{X}$ and $\tilde{\mathbf{X}}$ are neighbors if they differ on at most one column. We want algorithms that are ϵ-differentially private with respect to neighboring data matrices. Formally, an algorithm $\mathcal{A}$ is ϵ-differentially private if for every pair of neighboring data matrices $\mathbf{X}$ and $\tilde{\mathbf{X}}$ and every set $\mathcal{O}$ of possible outcomes, we have:

$$\Pr(\mathcal{A}(\mathbf{X}) \in \mathcal{O}) \leq e^\epsilon \Pr(\mathcal{A}(\tilde{\mathbf{X}}) \in \mathcal{O}) \,. \tag{1}$$

2 Iterative Eigenvalue Sampling for Covariance Estimation

The high-level approach of our method is to approximate the eigendecomposition of the covariance matrix $\mathbf{C}$ by privately estimating the collection of eigenvalues and eigenvectors separately. Since the vector of eigenvalues has low sensitivity, we can accurately estimate them using the Laplace mechanism. To estimate the eigenvectors we use the exponential mechanism to sample a direction from the unit sphere $\mathcal{S}^{d-1}$ that approximately maximizes $\theta^\top \mathbf{C}\theta$, which measures the directional variance of C captured by the direction θ. We constrain each sampled eigenvector to be orthogonal to the vectors sampled so far. Our main contribution is to provide utility guarantees for this method. Based on this utility guarantee, we are able to divide a total privacy budget of ϵ across the $d+1$ private queries made by the algorithm in order to optimize accuracy of the final output. Pseudocode for this approach is given in Algorithm 1

Algorithm 1 Iterative Eigenvector Sampling

Input: True covariance matrix $\mathbf{C} = \mathbf{X}\mathbf{X}^\top \in \mathbb{R}^{d \times d}$, privacy parameters $\epsilon_0, \ldots, \epsilon_d$.

1. Let $\hat{\lambda}_i = \lambda_i(\mathbf{C}) + \mathrm{Lap}(\epsilon_0/2)$ for $i = 1, \ldots, d$.
2. Let $\mathbf{C}_1 = \mathbf{C}$.
3. Let $\mathbf{P}_1 = \mathbf{I}$
4. For $i = 1, \ldots, d$:
   (a) Sample $\hat{u}_i$ from density proportional to $f_{\mathbf{C}_i}(u) = \exp(\frac{\epsilon_i}{4} u^\top \mathbf{C}_i u)$ supported on $\mathcal{S}^{d-1}$ and let $\hat{\theta}_i = \mathbf{P}_i^\top u_i$
   (b) Let $\mathbf{U}_i$ be the subspace spanned by $\hat{\theta}_1, \ldots, \hat{\theta}_i$ and $\mathbf{P}_i \in \mathbb{R}^{(d-i) \times d}$ be the euclidean projection matrix onto the orthogonal complement of $\mathbf{U}_i$.
   (c) Let $\mathbf{C}_{i+1} = \mathbf{P}_i \mathbf{C} \mathbf{P}_i^\top$.
5. Output $\hat{\mathbf{C}} = \sum_{i=1}^d \hat{\lambda}_i \hat{\theta}_i \hat{\theta}_i^\top$.

This method applies the Laplace mechanism to estimate the vector of eigenvalues of C, and the exponential mechanism to estimate each eigenvector. Our privacy guarantees follow from bounds on the sensitivity of the vector of eigenvalues of the covariance matrix C, and bounds on the sensitivity of the utility function used with the exponential mechanism on each iteration.

Theorem 1. *Algorithm 1 preserves ϵ-differential privacy, where $\epsilon = \sum_{i=0}^d \epsilon_i$.*

Proof sketch. We show that the sensitivity of the vector-valued query of all d eigenvalues has ℓ_1-sensitivity at most 2. This implies that we can estimate all d eigenvalues while preserving ϵ_0-differential privacy by adding $\mathrm{Lap}(2/\epsilon_0)$ to each eigenvalue.

The approximate eigenvectors $\hat{u}_i$ are evaluated using the exponential mechanism applied to maximizing the function $\theta^\top \mapsto \theta \mathbf{C}_i \theta$ over the unit sphere. We show that for any fixed vector $\theta \in \mathcal{S}^{d-1}$, the maximum change in $\theta^\top \mathbf{C}_i \theta$ on two neighboring data matrices is at most 2. This implies that at round i, we preserve ϵ_i-differential privacy when sampling $\hat{u}_i$ using the exponential mechanism.

The final privacy guarantee follows from basic composition over the $d+1$ steps of the algorithm. □

Our main contribution is a utility guarantee for this algorithm bounding the Frobenius distance between $\hat{\mathbf{C}}$ and the true covariance matrix $\mathbf{C}$ as a function of the privacy parameters used for each step. An important consequence of this analysis is that we can optimize the allocation of our total privacy budget ϵ among the $d+1$ queries in order to get the best bound.

Theorem 2. *Let $\delta > 0$ and $\hat{\mathbf{C}}$ be the estimated covariance matrix with estimated eigenvectors $\hat{\theta}_1, \ldots, \hat{\theta}_d$ and eigenvalues $\hat{\lambda}_1, \ldots, \hat{\lambda}_d$. Then with probability at least $1-\delta$, we have*

$$\|\hat{\mathbf{C}} - \mathbf{C}\|_F \leq \widetilde{O}\left(\frac{d}{\epsilon_0} + \sqrt{\sum_{i=1}^{d} \frac{d}{\epsilon_i}\left(\frac{1}{\epsilon_0} + \hat{\lambda}_i\right)}\right)$$

where the $\widetilde{O}$ notation hides a logarithmic dependency on $1/\delta$.

Proof sketch. Our analysis has two main steps. First, we argue that at every iteration, the matrix $\hat{\theta}_i$ captures almost as much variance of $\mathbf{C}_i$ as the true top eigenvector of $\mathbf{C}_i$. Intuitively, this follows from the standard utility analysis of the exponential mechanism, together with a lower bound on the surface area of directions on $\mathcal{S}^{d-1}$ of approximately variance maximizing directions. Formally, we show that with probability at least $1-\delta$ over $\hat{\theta}$ sampled from the density proportional to $\exp(\frac{\epsilon}{4}\theta^\top \mathbf{C}\theta)$, we have

$$\hat{\theta}\mathbf{C}\hat{\theta}^\top \geq \max_{\theta \in \mathcal{S}^{d-1}} \theta \mathbf{C} \theta^\top - \tilde{O}(d/\epsilon). \tag{2}$$

Notice that from the definiton of $\hat{\theta}_i$ as $\hat{\theta}_i = \mathbf{P}_i u_i$ then $\hat{\theta}_i \mathbf{C} \hat{\theta}_i = u_i \mathbf{C}_i u_i$ and thus is approximately equal to $\lambda_{\max}(\mathbf{C}_i)$,the largest eigenvalue of $\mathbf{C}_i$. The crux of our proof is showing that $\lambda_{\max}(\mathbf{C}_i) \geq \lambda_i$. In other words, our sampling mechanism does not accumulate error in the projection steps. To see that the statement is true notice that if we made a mistake in sampling $\hat{\theta}_1$ then the projection $\mathbf{P}_2$ would keep some of the variance in the direction of the true eigenvector θ_1. Therefore $\lambda_{\max}(C_2)$ must include some contribution of λ_1. This implies that $\lambda_{\max}(\mathbf{C}_2) \geq \lambda_2$. The argument can be generalized to all the other sampled eigenvectors. We can now bound the squared Frobenius error as follows:

$$\|\hat{\mathbf{C}} - \mathbf{C}\|_F^2 \leq 2 \sum_{i=1}^{d} \lambda_i(\mathbf{C}) \cdot \left(\lambda_i(\mathbf{C}) - \hat{\theta}_i^\top \mathbf{C} \hat{\theta}_i\right).$$

But from equation (2) and the above discussion we know that $\lambda_i(\mathbf{C}) - \hat{\theta}_i^\top \mathbf{C} \hat{\theta}_i = \tilde{O}(d/\epsilon)$ which concludes the proof.

□

An important observation is that the utility guarantee given by Theorem 2 is expressed in terms of quantities that are known after estimating the eigenvectors. This allows us to fine-tune the division of our total privacy budget ϵ among the remaining eigenvector sampling tasks. In particular, we find that ϵ_i should be set proportionally to $\sqrt{1/\epsilon_0 + \hat{\lambda}_i}$.

3 Experiments

In this section we compare our algorithm against three other baselines:

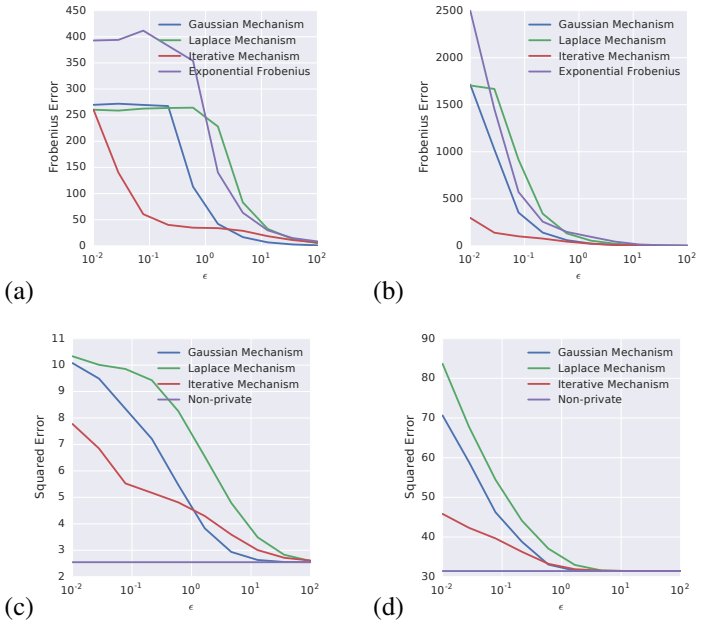

Figure 1: Results on real-world datasets. (a) & (b) Frobenius error of the released covariance matrix for the Wine and Airfoil data sets, respectively. (c) & (d) Average ridge regression error for the Wine and Airfoil data sets, respectively. The exponential mechanism was excluded here due to extremely large errors.

- The Laplace mechanism, which returns the matrix $\widehat{\mathbf{C}_L} = \mathbf{C} + \mathbf{L}$, where $\mathbf{L}$ is symmetric with each entry above the diagonal distributed as $\mathrm{Lap}(\frac{2d}{\epsilon})$.
- The Gaussian mechanism, which returns the matrix $\widehat{\mathbf{C}_N} = \mathbf{C} + \mathbf{N}$, where $\mathbf{N}$ is symmetric with each entry above the diagonal distributed as a Gaussian with mean 0 and standard deviation $\frac{\sqrt{2\log(1.25/\delta)}}{\epsilon}$. The parameter δ was set to 10^{-10} to make the algorithm roughly comparable to the other methods, which all achieve $(\epsilon, 0)$-differential privacy.
- The exponential mechanism, which samples a matrix $\widehat{\mathbf{C}}$ with probability proportional to $f(\widehat{\mathbf{C}}) = \exp(\|\widehat{\mathbf{C}} - \mathbf{C}\|_F)$.

We measured the performance of our algorithm using two metrics: the Frobenius distance to the true covariance matrix, and the error of ridge regression when evaluated on the original dataset. For the second metric we predicted each column in the data matrix as a function of the other columns, illustrating that the covariance matrix can be used for a variety of learning tasks. (This is in contrast to methods for directly performing differentially private regression on raw data, which need to use additional privacy budget for each regression.) The result shown is the average error across all target columns.

The data sets used for this experiment were the Wine data set, consisting of 200 examples and 14 dimensions, and the Airfoil data set, consisting of 1500 examples and 6 dimensions. The privacy parameters of each stage our algorithm were tuned according to to the discussion following Theorem 2 (while maintaining a total privacy cost of ϵ). We ran each experiment 100 times; the results are shown in Figure 1. Notice that for almost all choices of ϵ, our algorithm outperforms the others. This seems to indicate that there is a benefit to adding noise inversely proportional to the importance of each eigenvector.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318. ACM, 2016.

K. Chaudhuri, A. Sarwate, and K. Sinha. Near-optimal algorithms for differentially-private principal component analysis. In *NIPS*, 2012.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014a.

Cynthia Dwork, Kunal Talwar, Abhradeep Thakurta, and Li Zhang. Analyze gauss: optimal bounds for privacy-preserving principal component analysis. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 11–20. ACM, 2014b.

Prateek Jain, Pravesh Kothari, and Abhradeep Thakurta. Differentially private online learning. In *Conference on Learning Theory*, pages 24–1, 2012.

Frank McSherry and Ilya Mironov. Differentially private recommender systems: Building privacy into the netflix prize contenders. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 627–636. ACM, 2009.