

Towards Real-time Performance of Data Value Hiding for Frequent Data Updates by Incremental Matrix Decomposition

Jie Wang^{*}, Jun Zhang[†]

Department of Computer Science
University of Kentucky, Lexington, KY 40506-0046
jwanga@csr.uky.edu, jzhang@cs.uky.edu

Justin Zhan

The Heinz School
Carnegie Mellon University
justinz@andrew.cmu.edu

Abstract

Hiding data values in privacy-preserving data mining (PPDM) protects information against unauthorized attacks while maintaining analytical data properties. The most popular models are designed for constant data environments. They are usually computationally expensive for large data sizes and have poor real-time performance on frequent data growth. Considering that updates and growth of source data are becoming more and more popular in on-line environments, a PPDM model that has quick responses on the data updates in real-time is appealing. To increase the speed and response of the singular value decomposition (SVD) based model, we have applied an improved incremental SVD-updating algorithm. The performance and effectiveness of the improved algorithm have been examined on synthetic and real data sets. Experimental results indicate that the introduction of the incremental matrix decomposition produces a significant increase in speed for the SVD-based data value hiding method, better scalability, and better real-time performance of the model, thereafter. It also provides potential support for the use of the SVD technique in the On-Line Analytical Processing for business data analysis.

1. Introduction

Data use can be classified in five ways: data distribution, data modification, data mining algorithms, data or rule hiding, and privacy preservation [18]. Privacy preserving data mining (PPDM) has been studied for several years [2], starting with [5]’s seminal publication on the topic in 1996.

^{*}URL: <http://www.csr.uky.edu/~jwanga>

[†]URL: <http://www.cs.uky.edu/~jzhang>. The research work of the authors was supported in part by NSF under grant CCF-0527967, in part by KSEF under grant KSEF-148-502-06-186, in part by Alzheimer’s Association under grant NIGR-06-25460, and in part by NIH under grant 1R01HL086644-01.

PPDM can be viewed as a method of computer security with the advantage of handling attacks originating from the inside of organizations.

For the attacks from outside sources, access control mechanisms can be used to assign different levels of rights to different users in order to control data disclosure. For public access, only the non-confidential attributes in the databases are published to the partners or the public. However, when the threat comes from inside, the problem becomes more complicated. It has been reported by the Wall Street Journal in February 2006 that companies were finding that insiders pose as great a risk to computer security as outside attackers. Among all traceable attacks from 1999 to 2005, 40% were internal [16].

By using data analysis tools including data mining methods, some access rights can be given to employers for performing analysis of the data. At this stage, data privacy would be out of control without any data preprocessing. Thus, in the absence of adequate safeguards, the use of data mining can jeopardize privacy. Obtaining the potential benefits of data mining with privacy-sensitive technologies can enable wider social acceptance of many new services and promote applications based on knowledge discovery.

A variety of methods have been designed for preserving the privacy of sensitive or confidential information while still maintaining the original analytical properties of the data in the application of data mining, which is referred to as *data value hiding* in this paper. This can be divided by data set types (numerical-valued data vs. categorical-valued data or mixed-type data), data location (centralized data vs. distributed data), or data mining methods (classification, clustering, association rule mining, etc.). Some of the most popular methods include the k -anonymity model [15], the noise-additive model [10, 1, 4, 7], the random projection model [12, 13], the matrix-decomposition-based model [20, 19, 21], and secure multiparty computation [23].

These different methods have been designed to adapt to specific applications such as the matrix-decomposition-based model that has demonstrated its great potential for

data pattern maintenance in numerical-valued data sets [20, 19, 21]. Singular value decomposition (SVD) and nonnegative matrix factorization (NMF) are two techniques used in this model.

Computational cost has not traditionally been emphasized in previous work on PPDM. The data source may change or new data elements may be added, like in situations of financial transaction streams and network activity streams. Another scenario is that the data source is in an online setting where data must be incorporated into the data value hiding model as it arrives. The data value hiding model is required to be updated in real-time. We know that matrix operations are the core of implementation in the random projection model and the matrix-decomposition-based model. Eventually, the computational performance of these two models are subjected to the size of the data sets. If the data is frequently updated with increasing size, the computation of new models at each time would incur a sizable delay. It is important to figure out how to adjust the models dynamically for a real-time response when dealing with changes to the data matrix.

Ignoring effectiveness, a good PPDM model should be computationally economical and practically robust for constant and dynamical data sources. First, it should be scalable and computationally applicable to high-dimensional data. Secondly, it should be adaptive to the external perturbations, including the addition of new data, the removal of old data and so on. Considering that data streaming is becoming more and more popular in online environments, it is required that a good PPDM model make a quick response to external perturbations and produce a new solution in real time.

Therefore, we will explore the computational needs of PPDM algorithms so as to handle growth and change in data sources. The work in this paper focuses upon improving the real-time performance of the existing matrix-decomposition-based data value hiding model on a frequently-updated data source. In §2, the problem is described and then followed by an introduction of the related work. A performance improvement analysis is presented in §3, designed especially for the thin SVD-based data value hiding method. In §4, an improved SVD updating algorithm is proposed and its performance evaluation is made in §5. Finally, our work is summarized in §6.

2 Problem Description and Related Work

In this paper, the context of the problem is specified as a data set that is subjected to frequent additions of new elements. Let $A \in \mathbb{R}^{n \times m}$ be a matrix representation of a data set containing n subjects and m attributes. Let $\tilde{A} \in \mathbb{R}^{n \times m}$ be a matrix representation of the distorted data set that is generated from A using data value hiding methods. In or-

der to correspond to the two kinds of elements in a typical data set (subject and attribute), an update of A , due to an addition, results in two kinds of augmentations: row/subject augmentation and column/attribute augmentation.

In this context, the targeted solution \tilde{A} is one of the solutions for the general purposes of data value hiding, which should be a real-time solution that is adaptive to frequent updates of the data elements and is scalable to the increasing size of the data set A . The following is a brief description of the two data value hiding models.

2.1 Random Projection Model

This model is mostly multiplicative perturbation in the context of computing the inner product matrix [12, 13]. Let $R \in \mathbb{R}^{m \times k}$ be a matrix generated with entries randomly chosen from a given distribution $\mathcal{N}(0, \sigma_r)$ with zero mean, $\mu = 0$ and variance σ_r , across columns, we have

$$\tilde{A} = AR \quad (1)$$

for right multiplication. For left multiplication, it becomes

$$\tilde{A} = RA. \quad (2)$$

In the random projection-based method, let $k = m$, since the dimension size should be maintained.

If R is nonorthogonal, according to the Johnson Lindenstrauss Lemma [9], the Euclidean distance is approximated on expectations up to a constant factor, and the random projection methods may suffer from the loss of Euclidean distances due to the nonorthogonal matrix R . We denote this method as *Arp* and *rpA*. The computational complexity is due to a matrix multiplication and is of the order $\mathcal{O}(nmm)$, and if A is sparse with about c nonzero entries per row, the complexity is of the order $\mathcal{O}(cnm)$.

If R is orthogonal, then the projection exactly preserves the inner product of A , which is the Euclidean distance.

$$\tilde{A}\tilde{A}^T = ARR^T A^T = AA^T. \quad (3)$$

We denote this by *Arpo* and *rpOA*. The complexity will be increased with the cost incurred by orthogonalizing R , which is in the order of $\mathcal{O}(n^3)$. *Arpo* is of the order $\mathcal{O}(nm^2 + n^3)$ and is always computationally expensive.

2.2 Matrix-decomposition-based Model

In a matrix-decomposition-based model, instead of an addition or multiplication of the external random noise, a low-ranking approximation is used as the distorted data set \tilde{A} , which is computed by conducting matrix decomposition techniques on the original data set A . If the *complete SVD* is used, then

$$A = U\Sigma V^T, \quad (4)$$

Table 1. A comparison of four data value hiding methods on computation time.

Methods	the source matrix: 3000 × 3000				
	NMF-based	thin SVD-based	<i>Arp</i>	<i>Arpo</i>	complete SVD
CPU time (s)	7.0501	124.3388	33.2897	325.9687	553.3256
Parameter	$K = 100$	$K = 100$	$\mathcal{N}(0, 1)$	$\mathcal{N}(0, 1)$	

where $U \in \mathbb{R}^{n \times n}$, Σ is a diagonal matrix of size $n \times m$, having only r nonzero entries (the singular values of A) as its diagonal entries are in descending order, and $V \in \mathbb{R}^{m \times m}$. U and V are orthogonal eigenvectors associated with the r nonzero eigenvalues of AA^T and $A^T A$. The computation cost is $\mathcal{O}(nm^2 + n^2m + m^3)$.

If the first K nonzero singular values are considered and the last $(r - K)$ nonzero singular values are truncated from Σ , then

$$A = A^{(K)} + E_K, \quad (5)$$

where $A^{(K)}$ is a rank- K approximation of A ,

$$A^{(K)} = U_{:(1:K)} \Sigma_K V_{:(1:K)}^T. \quad (6)$$

E_K is the approximation error of $A^{(K)}$, $E_K = U_{:(K+1:r)} \Sigma_{r-K} V_{:(K+1:r)}^T$.

Let

$$\tilde{A} = A^{(K)} \quad (7)$$

This is referred to as the *thin SVD-based method* (which is a SVD-based method in [20]). It has a computation cost of $\mathcal{O}(n^2K + nK^2 + K^3)$ and has an increased speed compared to the complete SVD. It has been proven that the distance between A and A^K is the minimum in the sense of the Frobenius norm in all the rank- K approximations of A [6].

Another matrix decomposition used in this model is the nonnegative matrix factorization [14]. For a given nonnegative $A \in \mathbb{R}_+^{n \times m}$, and a pre-specified positive integer $K \leq \min\{n, m\}$, there are two nonnegative matrices: H and W , *s.t.*, $A \approx HW$ that minimizes the objective function

$$f(H, W) = \frac{1}{2} \|A - HW\|_F^2, \quad (8)$$

where $H \in \mathbb{R}_+^{n \times K}$ and $W \in \mathbb{R}_+^{K \times m}$. If the distorted data is taken as

$$\tilde{A} = HW, \quad (9)$$

we refer it as *NMF-based method* [21]. An efficient algorithm of NMF is alternating nonnegative least-squares using projected gradients in [11] and its overall computation cost is $\text{iter} \times \mathcal{O}(nmK) + \text{subIter} \times \mathcal{O}(tmK^2 + tnK^2)$, where iter and subIter are the number of the two iterations.

3 Performance Improvement Analysis on Thin SVD-based Model

Basically, a reduction of computation time for the model provides an increase in speed on the model response time with data updates. Before we discuss possible solutions, it is helpful to look at Table 1: a simple comparison of the computation time of four data hiding methods on a 3000 × 3000 matrix: thin SVD-based, NMF-based, *Arp*, *Arpo*. It was conducted in MATLAB 7.1. The absolute time does not have much meaning (as it is machine-dependent), however, the relative difference on the running time would imply an ordering of the speed of these four methods. In Table 1, it is observable that the NMF-based model is significantly faster than the other three methods, with a running time of only 7 seconds. *Arp* places second. However, *Arpo* is very expensive, computationally, due to its orthogonization operation, while the thin SVD-based model runs much faster than *Arpo* partly because a part of submatrices is used in computation instead of the complete submatrices in the complete SVD. By using thin SVD instead of the complete SVD, the running time is significantly decreased. Refer to Table 1 to see that the CPU time for the complete SVD is 553.3256 seconds, while the thin SVD only takes 124.3388 seconds. However, compared to the NMF-based model and the *Arp* model, some improvement is still required for the thin SVD-based method.

If the data set A is subjected to frequent element additions, and at each time, a new distorted data \tilde{A} is needed to be computed repeatedly on the new updated data set, then the thin SVD-based method and *Arpo* are not scalable. In this paper, we attempt to speed up the thin SVD-based model since the thin SVD-based method experimentally demonstrates a competitive data mining accuracy compared to random projection model as shown in Table 2, which is a comparison between the two models by conducting \mathcal{K} -means clustering and classification by SVMlight [8] on Wisconsin Diagnostic Breast Cancer Database (WDBC) [3]. WDBC contains 569 subjects and 30 real attributes. 357 subjects are in the group of benign, and 212 are in the malignant group. Its best known classification accuracy is 97.5% using 10-fold cross validation [3]. RE is the relative error between A and \tilde{A}

$$\text{RE} = \frac{\|A - \tilde{A}\|_F}{\|A\|_F}. \quad (10)$$

Table 2. Accuracy comparison of five methods for WDBC.

Methods	RE	Parameter	K-means %	SVMLight %
thinSVD	0.0054	$K=4$	91.7399	96.1300
<i>Arp</i>	0.9721	$\sigma_r=0.1109$	85.2373	95.0791
<i>Arpo</i>	1.0727	$\sigma_r=5.8627$	84.3585	93.6731
<i>rpA</i>	1.0255	$\sigma_r=0.0100$	50.9666	51.1424
<i>rpoA</i>	1.3417	$\sigma_r=1.4227$	52.5483	53.9543

The thin SVD-based model consists of matrix decompositions primarily from the SVD computation. Even though the algorithm is extremely stable, computing a full SVD is a problem of the order of $\mathcal{O}(nm^2 + n^2m + m^3)$ for a matrix size of n by m . All the data must be processed at one time, and the computation time increases exponentially with the addition of new subjects into the databases.

The intuitive choice is to only modify the old SVD model to reflect the addition of the new data records, not to recompute the SVD of the new full data matrix. In the next section, we will introduce an improved incremental SVD updating algorithm to enhance the performance of the thin SVD-based data hiding method.

4 Improved Incremental SVD Updating Algorithms

The improved incremental SVD algorithm is based on the updating methods introduced in [22, 17]. This method requires one QR decomposition and one SVD per update. However, these potentially expensive computations are performed on small intermediate matrices, where the computational complexity depends on the size of the update and/or the reduced dimension K , but not on the size of the original data matrix. Depending on subject/attribute addition, there are two updating algorithms: subject-updating and attribute-updating. Essentially, our improved incremental SVD algorithm is based on the algorithms in [17].

4.1 Updating Subjects

Let $A \in \mathbb{R}^{n \times m}$ be the original data matrix, and $A = [A0; T]$, where $A0 \in \mathbb{R}^{t \times m}$ and $T \in \mathbb{R}^{q \times m}$ with q is the number of new subjects to be appended, and $n = t + q$.

$$\begin{bmatrix} A0 \\ T \end{bmatrix} \longrightarrow A. \quad (11)$$

Assuming the rank- K SVD of $A0$ is known in advance,

$$A0^{(K)} = U_{\cdot(1:K)} \Sigma_K V_{(1:K)}^T.$$

For simplicity, we use U_K for $U_{\cdot(1:K)}$, and V_K for $V_{(1:K)}$ in the following. The purpose of the algorithm is to modify the SVD of $A0$ based on the new data, T .

Let $\hat{T} \in \mathbb{R}^{m \times q}$ and

$$\hat{T} = (I_m - V_K V_K^T) T^T. \quad (12)$$

Perform the QR decomposition of \hat{T} , $Q_T R_T = \hat{T}$, where $Q_T \in \mathbb{R}^{m \times q}$ is orthonormal, and $R_T \in \mathbb{R}^{q \times q}$ is upper triangular. Then

$$\begin{aligned} A &= \begin{bmatrix} A0 \\ T \end{bmatrix} \approx \begin{bmatrix} A0^{(K)} \\ T \end{bmatrix} \\ &= \begin{bmatrix} U_K & \mathbf{0} \\ \mathbf{0} & I_q \end{bmatrix} \begin{bmatrix} \Sigma_K & \mathbf{0} \\ TV_K & R_T^T \end{bmatrix} [V_K \quad Q_T]^T. \end{aligned} \quad (13)$$

Now let $\hat{A} \in \mathbb{R}^{(K+q) \times (K+q)}$ be the matrix defined by

$$\hat{A} = \begin{bmatrix} \Sigma_K & \mathbf{0} \\ TV_K & R_T^T \end{bmatrix}. \quad (14)$$

In [17], a complete SVD of \hat{A} is computed. Here, a small improvement is made and a rank- K approximation of \hat{A} is computed instead.

$$\hat{A} \approx \hat{U}_K \hat{\Sigma}_K \hat{V}_K^T \quad (15)$$

where $\hat{U}_K \in \mathbb{R}^{(K+q) \times K}$, $\hat{V}_K \in \mathbb{R}^{(K+q) \times K}$ and $\hat{\Sigma}_K \in \mathbb{R}^{K \times K}$. Then the thin SVD of A in K dimensions is

$$A^{(K)} = \left(\begin{bmatrix} U_K & \mathbf{0} \\ \mathbf{0} & I_q \end{bmatrix} \hat{U}_K \right) \hat{\Sigma}_K \left([V_K \quad Q_T] \hat{V}_K \right)^T. \quad (16)$$

This procedure has a computational complexity of $\mathcal{O}(K^3 + (m+t)K^2 + (m+t)Kq + q^3)$ [17].

4.2 Updating Attributes

Let $A \in \mathbb{R}^{n \times m}$ be the original data matrix, and $A = [A0, F]$, where $A0 \in \mathbb{R}^{n \times t}$ and $T \in \mathbb{R}^{n \times p}$ with q is the number of new attributes to be appended, and $m = t + p$.

$$[A0 \quad T] \longrightarrow A. \quad (17)$$

Let $\hat{F} \in \mathbb{R}^{n \times p}$ and

$$\hat{F} = (I_n - U_K U_K^T) F^T. \quad (18)$$

Perform the QR decomposition of \hat{F} , $Q_F R_F = \hat{F}$, where $Q_F \in \mathbb{R}^{n \times p}$ is orthonormal, and $R_F \in \mathbb{R}^{p \times p}$ is upper triangular. Then

$$\begin{aligned} A &= [A0 \quad F] \approx [A0^{(K)} \quad F] \\ &= \begin{bmatrix} U_K & Q_F \end{bmatrix} \begin{bmatrix} \Sigma_K & U_K^T F \\ \mathbf{0} & R_F \end{bmatrix} \begin{bmatrix} V_K^T & \mathbf{0} \\ \mathbf{0} & I_p \end{bmatrix}. \end{aligned} \quad (19)$$

Table 3. Run time and RE of two SVD algorithms.

Rows	Incremental thin SVD		Lanczos thin SVD	
	Run time(s)	RE	Run time(s)	RE
3000	218.7799	0.2729	242.9899	0.2720
4000	233.3299	0.2747	321.7100	0.2732
5000	228.0000	0.2758	396.6999	0.2740
6000	231.5399	0.2762	475.7899	0.2742
7000	242.0900	0.2764	568.7299	0.2743
8000	245.0100	0.2767	735.2900	0.2745
9000	244.5699	0.2772	736.9499	0.2749
10000	257.4699	0.2772	825.7900	0.2748

Now let $\hat{A} \in \mathbb{R}^{(K+p) \times (K+p)}$ be the matrix defined by

$$\hat{A} = \begin{bmatrix} \Sigma_K & U_K^T F \\ \mathbf{0} & R_F \end{bmatrix}, \quad (20)$$

we do the same improvement as updating subjects in (15),

$$\bar{A} \approx \bar{U}_K \bar{\Sigma}_K \bar{V}_K^T \quad (21)$$

where $\bar{U}_K \in \mathbb{R}^{(K+p) \times K}$, $\bar{V}_K \in \mathbb{R}^{(K+p) \times K}$ and $\bar{\Sigma}_K \in \mathbb{R}^{K \times K}$. Then the thin SVD of A in K dimensions is

$$A^{(K)} = ([U_K \quad Q_F] \bar{U}_K)^T \bar{\Sigma}_K \begin{bmatrix} V_K & \mathbf{0} \\ \mathbf{0} & I_p \end{bmatrix} \bar{V}_K \quad (22)$$

This procedure has a computation complexity of $\mathcal{O}(K^3 + (m+t)K^2 + (m+t)Kp + p^3)$ [17].

5 Experiments and Results

Several experiments were conducted in MATLAB 7.1 on synthetic data sets and real data sets to compare the run time, relative error and data mining accuracy between Lanczos SVD and the improved incremental thin SVD as described in §4.

5.1 Subject/Row Updating by Incremental Thin SVD

In this experiment, the incremental thin SVD is examined by adding new subjects. The data set is a synthetic real-value matrix of size of 10000×1000 with the rank of 100. The rank of approximation in the thin SVD is set up to 60. The starting matrix consists of the first 2000 subjects. The rest of the 8000 subjects are repetitively added to the starting matrix for 8 times. At each step, 1000 new subjects are added and a new rank-60 thin SVD are computed by two algorithms: Lanczos SVD and incremental

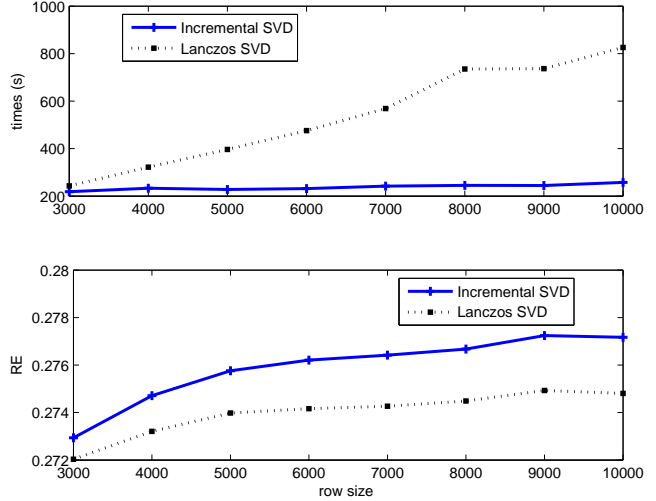


Figure 1. Run time and RE of incremental SVD updating (solid line) versus Lanczos SVD (dashed line), as a function of a repetitive addition of 1000 rows for 8 times, on a 10000×1000 random matrix and its rank is 100. The upper figure shows the run time of each addition. The lower figure shows RE.

SVD. The experimental results are listed in Table 3 and are plotted in Figure 1. The relative/approximation error here is defined in (10) as RE. If at each step, the augmentation size is 1000, then the run time of the incremental SVD based on the old SVD approximation is much less than that of the Lanczos SVD. At the same time, there is not much effect on the approximation error. For example, if calculating the full matrix by the Lanczos SVD, it takes 825.79 seconds; if updating the SVD from the size of 9000×1000 , the cpu time is 257.47 seconds and only takes 31.18% of run time for the Lanczos thin SVD. Meanwhile, the relative error is 0.2772, which is very similar to 0.2748 by the Lanczos thin SVD.

5.2 Attribute/Column Updating by Incremental Thin SVD

A synthetic matrix of the size of 3000×3000 with the rank of 100 is randomly generated in order to examine the performance of attribute updating. The addition of the attributes/columns to the starting matrix of size 3000×80 is repeated 100 times with 22 columns each time. The rank of approximation is set to 80. The comparison is shown in Figure 2. For this data set, the advantages of incremental SVD are attractive, considering the cumulative CPU time for these 100 additions is only 29.1719 seconds, and at the

same time, the Lanczos SVD requires almost 104.7306 seconds. Moreover, the approximation errors are very close for the two methods.

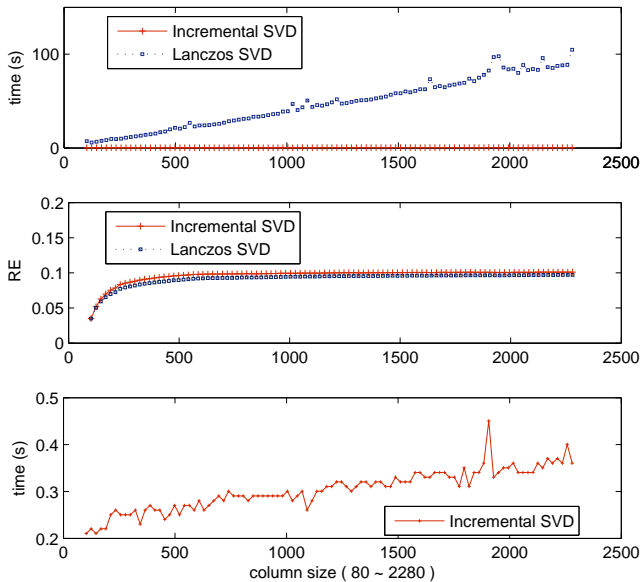


Figure 2. Run time and RE of incremental SVD updating (solid line) versus Lanczos SVD (dashed line), as a function of a repetitive addition of 22 columns for 100 times, on a 3000×3000 random matrix. Its rank is 100. The top figure shows the run time of each addition. The middle figure shows RE. The bottom figure is the amplified plot of the run time of the incremental SVD.

5.3 Performance Evaluation of the Incremental Thin SVD on WBC

In this experiment, the data mining accuracies are considered in the comparison and the real data set, and the WBC [3] database is used. WBC consists of 699 subjects and 10 integer-valued attributes. Some preprocessing is done, as in [20]. The experiment is designed as follows: the starting matrix is set up to the first 199 subjects/rows and the approximation rank in SVD is 7, then the rest of the 500 subjects are appended repeatedly by 50 rows each for 10 times. At each time, for both methods, a new rank-7 approximation is computed and its data mining accuracies are evaluated both on SVMlight classification and \mathcal{K} -means clustering. Figure 3 shows the comparison of run times and approximation errors of the two methods. It is shown that the time of each step in the incremental SVD

is less than the Lanczos SVD on the full data matrix. The difference of the two approximation errors is in the order of 0.001.

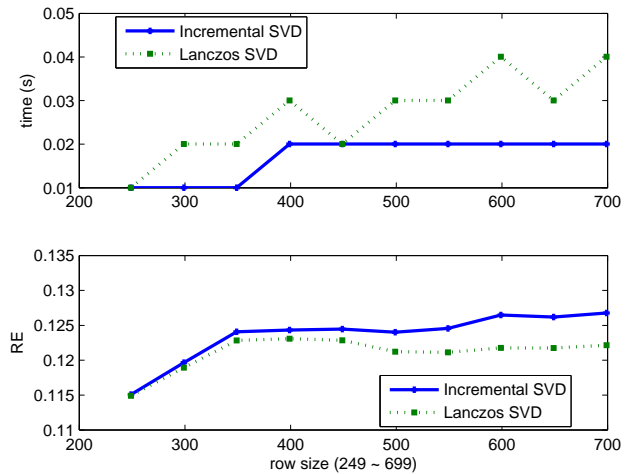


Figure 3. Run time and RE of incremental SVD updating (solid line) versus Lanczos (dashed line), as a function of a repetitive addition of 500 rows for 10 times, on WBC. The upper figure shows the run time of each addition. The lower figure shows the RE.

Secondly, by the two methods, the twenty data matrices with row numbers of 299 to 699 are tested in SVMlight classification. Figure 4 shows that the accuracies of this data are the same as their counterparts by other methods. It implies that incremental SVD does not introduce any observable effect on the classification accuracy. Thirdly, \mathcal{K} -means clustering is executed on the two rank-7 approximations. One is the rank-7 approximation by the Lanczos SVD of the original WBC and another is the rank-7 approximation by the incremental SVD, which is updated from 199 rows to 699 rows. We examine whether the incremental SVD will affect the clustering quality. Figure 5 shows the cluster distributions and Silhouette value for the two approximations of WBC.

In MATLAB 7.1, the Silhouette value, $s(i)$, is used as a measure of how similar the i th subject is to subjects in its own cluster compared to subjects in other clusters. It ranges from -1 to $+1$. It is defined in MATLAB 7.1 code as

$$s(i) = (\min(b(i, :), 2) - a(i)) ./ \max(a(i), \min(b(i, :), 2))$$

where $a(i)$ is the average distance from the i th point to the other points in its cluster, and $b(i, k)$ is the average distance from the i th point to the points in another cluster k . $./$ is a

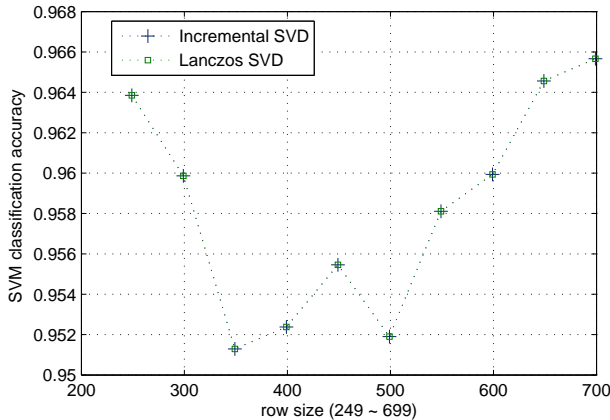


Figure 4. SVM classification accuracy of two rank-7 approximations as a function of a repetition addition of 50 rows. Two methods: incremental SVD updating (solid line) versus Lanczos (dashed line).

element-wise division. In this experiment, the row updating in calculating thin SVD does not have a negative influence on clustering.

6 Summary

This paper has presented an improved SVD-based data value hiding method. The decomposition is derived from updating the previous decomposition solution in an incremental way, instead of starting a new decomposition on the full data matrix. In our experiments, the increase in speed associated with this improved method is encouraging. More importantly, no distinctness from the traditional SVD-based method is found on the data mining results. This will allow us to address the real-time performance concern with the SVD-based method when a quick response performance is required for updates of large data size. In the meantime, this approach also provides possible support for the application of SVD in the On-Line Analytical Processing, which is essential in business data analysis featuring large amounts and frequent growth of data.

References

- [1] C. C. Aggarwal and P. S. Yu. A condensation approach to privacy preserving data mining. In *Advances in Database Technology - EDBT 2004, 9th International Conference on Extending Database Technology, Heraklion, Crete, Greece, March 14-18, 2004*, pages 183–199, 2004.
- [2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the ACM SIGMOD Conference on Man-*

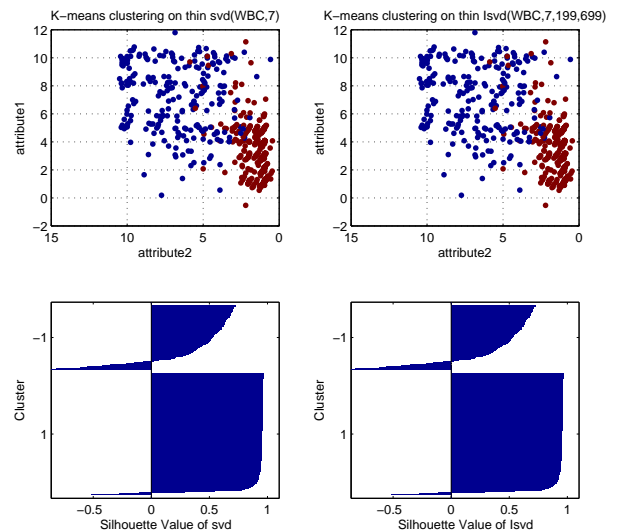


Figure 5. Cluster distribution and Silhouette Value of \mathcal{K} -means clustering on a rank-7 approximation of WBC, by Lanczos SVD and Incremental SVD, respectively. The two figures on the left are Cluster distribution and Silhouette Value using thin Lanczos SVD. The two figures on the right are cluster distribution and Silhouette Value using thin Incremental SVD, updated from 199 rows to 699 rows, and at each step increased by 50 rows.

agement of Data, pages 439–450, Dallas, Texas, May 2000. ACM Press.

- [3] A. Asuncion and D. Newman. UCI machine learning repository, 2007.
- [4] K. Chen and L. Liu. Privacy preserving data classification with rotation perturbation. In *ICDM '05: Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 589–592, Washington, DC, USA, 2005. IEEE Computer Society.
- [5] C. Clifton and D. Marks. Security and privacy implication of data mining. In *Proceedings of the Workshop on Data Mining and Knowledge Discovery*, number 96-08, pages 15–19, Montreal, Canada, June 1996. University of British Columbia Department of Computer Science.
- [6] C. Eckart and G. Young. The approximation of one matrix by another of low rank. *Psychometrika*, 1(3):211–218, 1936.
- [7] A. V. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *PODS*, pages 211–222, 2003.
- [8] T. Joachims. *Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning*. MIT-Press, 1999.

- [9] W. Johnson and J. Lindenstrauss. Extensions of Lipschitz mapping into Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.
- [10] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. Random-data perturbation techniques and privacy-preserving data mining. *Knowledge and Information System*, 7(4):387–414, 2005.
- [11] C. J. Lin. Projected gradient methods for nonnegative matrix factorization. *Neural Computation*, 19(10):2756–2779, October 2007.
- [12] K. Liu, H. Kargupta, and J. Ryan. Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):92–106, 2006.
- [13] K. Muralidhar and K. B. et al. Accessibility, security and accuracy in statistical databases: the case for multiplicative fixed data perturbation approach. *Management Science*, 9(4):1549–1564, 1995.
- [14] P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.
- [15] L. Sweeney. k-anonymity: a model for protecting privacy. *Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.
- [16] M. Totty. The dangers within. *The Wall Street Journal*, February 2006.
- [17] J. E. Tougas and R. J. Spiteri. Updating the partial singular value decomposition in latent semantic indexing. *Computational Statistics and Data Analysis*, 52:174–183, 2007.
- [18] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis. State-of-the-art in privacy preserving data mining. *SIGMOD Record*, 33(1):50–57, 2004.
- [19] J. Wang, J. Zhang, L. Liu, and D. Han. Simultaneous data and pattern hiding in unsupervised learning. In *The 7th IEEE International Conference on Data Mining - Workshops(ICDMW'07)*, pages 729–734, Omaha, NE, USA, October 2007. IEEE Computer Society.
- [20] J. Wang, J. Zhang, S. Xu, and W. Zhong. A novel data distortion approach via selective SSVD for privacy protection. *International Journal of Information and Computer Security*, 2(1):48–70, 2008.
- [21] J. Wang, W. Zhong, and J. Zhang. NNMF-based factorization techniques for high-accuracy privacy protection on non-negative-valued datasets. In *Proceedings of the 2006 IEEE Conference of Data Mining, International Workshop on Privacy Aspects of Data Mining*, pages 513–517. IEEE Computer Society, 2006.
- [22] H. Zha and H. D. Simon. On updating problems in latent semantic indexing. *SIAM J. Sci. Comput.*, 21(2):782–791, 1999.
- [23] J. Z. Zhan, L. Chang, and S. Matwin. Building k-nearest neighbor classifiers on vertically partitioned private data. In *Proceedings of the 2005 IEEE International Conference on Granular Computing (GrC)*, pages 708–711, 2005.