IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 6, 2018, accepted November 14, 2018, date of publication November 27, 2018,
date of current version December 27, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2883647

An Enhanced Incremental SVD Algorithm for Change Point Detection in Dynamic Networks

YONGSHENG CHENG[1], JIANG ZHU[2,3], (Member, IEEE), and XIAOKANG LIN[4]

[1]Beijing Institute of Space Long March Vehicle, Beijing 100076, China
[2]Ocean College, Zhejiang University, Zhoushan 316021, China
[3]Key Laboratory of Dynamic Cognitive System of Electromagnetic Spectrum Space, Nanjing University of Aeronautics and Astronautics, Ministry of Industry and Information Technology, Nanjing 211106, China
[4]Department of Electronic Engineering, Tsinghua University, Beijing 100084, China

Corresponding author: Jiang Zhu (jiangzhu16@zju.edu.cn)

This work was supported by the Zhejiang Province Science Foundation of China under Grant LQ18F010001.

ABSTRACT Change point detection is essential to understand the time-evolving structure of dynamic networks. Recent research shows that a latent semantic indexing (LSI)-based algorithm effectively detects the change points of a dynamic network. The LSI-based method involves a singular value decomposition (SVD) on the data matrix. In a dynamic scenario, recomputing the SVD of a large matrix each time new data arrives is prohibitively expensive and impractical. A more efficient approach is to incrementally update the decomposition. However, in the classical incremental SVD (incSVD) algorithm, the information of the newly added columns is not fully considered in updating the right singular space, resulting in an approximation error which cannot be ignored. This paper proposes an enhanced incSVD (EincSVD) algorithm, in which the right singular matrix is calculated in an alternative way. An adaptive EincSVD (AEincSVD) algorithm is also proposed to further reduce the computational complexity. Theoretical analysis proves that the approximation error of the EincSVD is smaller than that of the incSVD. Simulation results demonstrate that the EincSVD and the AEincSVD perform much better than the incSVD on change point detection, and the performance of the EincSVD is comparable to the batch SVD algorithm.

INDEX TERMS Change point detection, dynamic networks, incremental algorithm, singular value decomposition.

I. INTRODUCTION

Most real networks, including social networks, sensor networks, World Wide Web, and biological networks, etc., exhibit structural changes over time. These dynamic networks are always characterized as graph streams, where nodes represent individual objects, and edges represent relationships or interactions among these objects. The changes in the network are expressed by dynamic relationships among the nodes. Change point detection is an inherent problem in dynamic networks, where one needs to discover the unknown structures in the network, and detect their anomalous changes. As change points always correspond to anomalies in networks, detecting the change points is very helpful to understand the anomalous behaviours or faults in dynamic networks [1]–[4]. For example, in the public safety case, change point detection may assist to identify terrorist activities at an early stage [5]. In addition, the detection of change points is the first step to summarize the network activity with a fewer number of static graphs [6], which can be used to reduce the amount of data for representing a highly dynamic network.

Many statistical measures have been proposed to detect the change points of dynamic networks. Authors in [6] define an average distance between consecutive graph snapshots based on nodes' connectivity. A change point is declared when this distance exceeds some threshold. In [7], the authors consider an anomaly score based on the eigen decomposition of the adjacency matrix. Then a statistical test based on randomized power martingale is used to detect the change points. In [8], scan statistics, which capture the history of a node's neighborhood, are introduced to detect the change points. The work in [9] describes a parameter free method based on information theoretic principles to find the community structure and change points. A family of distances which can be tuned to quantify structural changes of dynamic networks was proposed in [10] and [11]. Although the above

detection methods are able to detect the changes of community structures, metrics and algorithms in [6], [7], [10], and [11] are designed for undirected networks, and algorithms in [8] and [9] are designed for unweighted networks. In dynamic social networks, a statistical infinite feature cascade approach to the anomaly detection problem is proposed in [12]. Yet, as the authors state, the paper merely studies directed networks. The work in [13] considers dynamic weighted directed graphs. The detection algorithm compares a simple similarity measure to a predefined threshold. However, as shown in [13], the performance is severely influenced by the choice of the threshold. In [14], a novel latent semantic indexing (LSI)-based approach is proposed, which is able to detect the change points in directed/undirected and weighted/unweighted dynamic networks. The LSI-based algorithm involves a singular value decomposition (SVD) of the edge-segment matrix. It is shown that the algorithm clearly identifies latent changes of the community structure and outperforms the detection method proposed in [13].

Despite of its good performance in detecting the change points, computing the SVD of a very large dataset is prohibitively expensive. Especially, in a dynamic scenario, the dataset is frequently updated with new columns or rows. Recomputing the batch SVD[1] of the matrix is impractical. In this case, incremental methods which compute the decomposition by successively updating the previous results would be preferred. In [15], folding-in is proposed as a computationally inexpensive approach for updating the SVD. It uses an existing SVD to represent the new information. However, the results produced by folding-in may deteriorate after only a small number of updates, because the new data has no effect on the decomposition of the original matrix. A more reliable approach, which is referred as incremental SVD (incSVD) in this work, is to incrementally update the partial SVD [16]. In this approach, the partial SVD of the updated matrix is obtained by modifying the decomposition of the original matrix. Compared with folding-in, the incSVD method is more accurate but more complex. As a trade-off of folding-in and incSVD, folding-up is proposed in [17]. The folding-up method switches from the process of folding-in to incSVD before the accuracy of the decomposition degrades significantly.

A. RELATED WORK AND MAIN CONTRIBUTION

As will be discussed in Section II-D, in incSVD, the orthogonal component of the added columns, which is perpendicular to the dominant component of previous columns, is not fully utilized in updating the right singular space. This leads to an inaccurate decomposition of the updated matrix. The resulting approximation error is more serious when the column space spanned by the new columns changes significantly from previous ones, which indicates an obvious change point. In addition, the error will accumulate in the updating process. To overcome this problem, an enhanced incSVD (EincSVD) algorithm is proposed in this work. In EincSVD, the right singular matrix is updated alternatively. More information on the newly added columns is captured in the right singular space. It is proved that the approximation error of EincSVD is smaller than that of incSVD. To reduce the complexity of EincSVD, an adaptive EincSVD (AEincSVD) algorithm, which combines incSVD and EincSVD, is also proposed. The advantages of EincSVD and AEincSVD over incSVD in detecting change points are verified through simulation results.

The main contributions of this work are summarized as follows,

- We propose an EincSVD algorithm, in which the right singular matrix is calculated in an alternative way. It is proved that the approximation error of EincSVD is smaller than that of incSVD.
- An AEincSVD algorithm is proposed as an adaptive combination of incSVD and EincSVD. The computation complexity of AEincSVD is lower than that of EincSVD. Meanwhile, AEincSVD can provide much better performance than incSVD in terms of approximation error and in detecting the change points of dynamic networks.
- Simulations on change point detection in dynamic networks are conducted. The effectiveness and efficiency of the proposed algorithms are verified via numerical experiments.

B. PAPER ORGANIZATION

The remainder of this paper is organized as follows. Related work and the motivation of this work are provided in Section II. In Section III, the EincSVD algorithm is proposed and the approximation error of EincSVD is proved to be smaller than that of incSVD. In addition, the AEincSVD algorithm is also proposed. In Section IV, the complexity of the related incremental SVD algorithms is analyzed and compared. In Section V, simulation results are provided. Finally, the conclusion goes in Section VI.

C. NOTATIONS

In the sequel, vectors are denoted by boldface lower-case letters, and matrices by boldface upper-case letters. For a matrix $\mathbf{A}$, let $\mathbf{A}^{\mathrm{T}}$ denote the transpose of $\mathbf{A}$. Let ran($\mathbf{A}$) denote the range (or column space) of $\mathbf{A}$. Vector $\mathbf{a}_i$ is the i-th column of $\mathbf{A}$. Let vec($\mathbf{A}$) denote the vector of columns of $\mathbf{A}$ stacked one under the other. Let $||\mathbf{A}||_{\mathrm{F}}$ denote the Frobenius norm of $\mathbf{A}$ and $\|\mathbf{a}_i\|_2$ be the l_2 norm of $\mathbf{a}_i$. The best rank-k approximation of $\mathbf{A}$ is denoted by $\mathbf{A}_k$, while $\mathbf{A}_k^{\mathrm{c}} = \mathbf{A} - \mathbf{A}_k$ is the complement of $\mathbf{A}_k$. By an abuse of notation, $\mathbf{S}_{\mathrm{A},k}$ denotes the diagonal matrix with only the largest k singular values of $\mathbf{A}$; $\mathbf{U}_{\mathrm{A},k}$ and $\mathbf{V}_{\mathrm{A},k}$ denote the left and right singular matrices with the corresponding k dominant singular vectors, respectively. Symbol $\sigma_k(\mathbf{A})$ denotes the k-th largest singular value of $\mathbf{A}$.

[1] Here, batch SVD refers to the standard SVD which calculates the decomposition from scratch with all the data.

II. PRELIMINARIES AND MOTIVATION

In this section, the LSI-based change point detection method [14] and the incSVD algorithm [16] are reviewed. Moreover, the motivation of proposing the EincSVD algorithm is also discussed.

A. LATENT SEMANTIC INDEXING (LSI)

LSI [18] is an important algorithm in the area of information retrieval. In LSI, the ij-th entry of the term-document matrix $\mathbf{A}$ corresponds to the term frequency of the i-th term in the j-th document. Terms are characterized by their appearance in the documents and documents are characterized by the terms that they contain. By performing a rank-k SVD on the term-document matrix, the dimensionality of the information retrieval problem is reduced. Terms and documents are represented by the row vectors of $\mathbf{U}_{\mathrm{A},k}$ and $\mathbf{V}_{\mathrm{A},k}$ in the latent space, respectively. The term-term, term-document, and document-document similarities are readily to compute. LSI has the potential to overcome the synonym and polysemy problems observed in traditional models. Another advantage of LSI is that relevant documents can be retrieved even when they do not match any query terms.

B. LSI-BASED CHANGE POINT DETECTION IN DYNAMIC NETWORKS

A dynamic network can be represented as a sequence of random graphs $\mathcal{G} = \{G^{(1)}, G^{(2)}, \cdots, G^{(T)}\}$, $T < \infty$. Each snapshot $G^{(t)} = (V^{(t)}, E^{(t)})$ is a static graph with vertex set $V^{(t)}$, edge set $E^{(t)}$ and adjacency matrix $\mathbf{A}^{(t)} = [A^{(t)}_{i,j}]$, where $A^{(t)}_{i,j}$ represents weight of the edge from vertex i to j. Without loss of generality, it is assumed that the vertex set $V^{(t)}$ is the same for all graph snapshots.

To track how the structure of the graphs $G^{(t)}, t \geq 1$, evolves over time, the consecutive timestamps are grouped into segments. Then a graph segment is defined as a set of consecutive graphs $\mathcal{G}^{(s)} = \{G^{(t_s)}, G^{(t_s+1)}, \cdots, G^{(t_{s+1}-1)}\}$, where $t_s < t_{s+1}$ [9]. For the s-th graph segment $\mathcal{G}^{(s)}$, we define an adjacency matrix $\mathbf{A}^s$, whose (i,j)-th element is $[\mathbf{A}^s]_{i,j} = \sum_{t=t_s}^{t_{s+1}-1} A^{(t)}_{i,j}$.

Let $\mathbf{e}^{(s)} = \mathrm{vec}(\mathbf{A}^s)$. That is, $\mathbf{e}^{(s)}$ is a vector containing the weights of all the edges in the s-th graph segment $\mathcal{G}^{(s)}$. For undirected graphs, $\mathbf{A}^s$ is a symmetric matrix. In this case, only the upper (or equivalently, the lower) triangular part of the adjacency matrix is needed to represent $\mathbf{A}^s$.

Then, the edge-segment matrix can be defined as $\mathbf{E} = [\mathbf{e}^{(1)}, \mathbf{e}^{(2)}, \cdots, \mathbf{e}^{(T_s)}]$, where T_s is the number of graph segments. For the purpose of a fair comparison between different graph segments, it is assumed that each segment is of equal length. The length can be chosen appropriately according to the size of data sets and the requirement of applications [3].

Similar to the term-document matrix in LSI, the (i,j)-th element $E_{i,j}$ of the above edge-segment matrix $\mathbf{E}$ can be viewed as the frequency that the i-th edge appears in the j-th graph segment [18]. As in LSI, by performing an SVD on $\mathbf{E}$, and projecting the decomposition into a lower dimensional latent space, the dissimilarity between two graph segments can be evaluated by the cosine distance between their low-dimensional representations. Explicitly, suppose the SVD of $\mathbf{E}$ is

$$\mathbf{E} = \mathbf{U}_{\mathrm{E}}\mathbf{S}_{\mathrm{E}}\mathbf{V}_{\mathrm{E}}^{\mathrm{T}}.$$

From the Eckhart-Young theorem [19], it is known that the rank-k SVD provides the best rank-k approximation of the matrix in the Frobenius norm sense. Mathematically, the rank-k approximation of $\mathbf{E}$ is expressed as

$$\mathbf{E}_k = \mathbf{U}_{\mathrm{E},k}\mathbf{S}_{\mathrm{E},k}\mathbf{V}_{\mathrm{E},k}^{\mathrm{T}}.$$

According to LSI [18], the graph segments are now represented by the column vectors of $\mathbf{S}_{\mathrm{E},k}\mathbf{V}_{\mathrm{E},k}^{\mathrm{T}}$. The relation between two consecutive graph segments can be characterized by the cosine distance between their corresponding k-dimensional representations,

$$d_{\cos}(i, i+1) = 1 - \frac{\mathbf{q}_i^{\mathrm{T}}\mathbf{q}_{i+1}}{\|\mathbf{q}_i\|_2\|\mathbf{q}_{i+1}\|_2}, \tag{1}$$

where $\mathbf{q}_i$ is the i-th column of $\mathbf{S}_{\mathrm{E},k}\mathbf{V}_{\mathrm{E},k}^{\mathrm{T}}$.

Intuitively, if the community structure of the dynamic network does not change much over time, consecutive graph segments will have similar descriptions and a small cosine distance. Whenever a graph segment changes severely compared to previous ones, the cosine distance is usually large. Based on the cosine distance between the consecutive graph segments, change points are detected by comparing the distance to an empirical threshold.

C. THE INCSVD ALGORITHM

In a dynamic scenario, new data arrives sequentially. In this case, rather than recomputing the decomposition from scratch each time the data matrix is updated, an incremental SVD algorithm would be preferred.

Let $\mathbf{E} \in \mathbb{R}^{m\times n}$ be the original edge-segment matrix. The SVD of $\mathbf{E}$ is $\mathbf{E} = \mathbf{U}_{\mathrm{E}}\mathbf{S}_{\mathrm{E}}\mathbf{V}_{\mathrm{E}}^{\mathrm{T}}$. Let $\mathbf{E}_k = \mathbf{U}_{\mathrm{E},k}\mathbf{S}_{\mathrm{E},k}\mathbf{V}_{\mathrm{E},k}^{\mathrm{T}}$ be the best rank-k approximation of $\mathbf{E}$. Assume that there are p new columns $\mathbf{D} \in \mathbb{R}^{m\times p}$ to be appended to $\mathbf{E}$, yielding a matrix $\mathbf{B} = [\mathbf{E}, \mathbf{D}]$. The goal is to calculate the rank-k SVD of $\mathbf{B}$ through updating methods.

Let $\tilde{\mathbf{D}} = (\mathbf{I} - \mathbf{U}_{\mathrm{E},k}\mathbf{U}_{\mathrm{E},k}^{\mathrm{T}})\mathbf{D}$. The columns of $\tilde{\mathbf{D}}$ are orthogonal to the subspace spanned by columns of $\mathbf{U}_{\mathrm{E},k}$. In general, $\mathrm{rank}(\mathbf{E}_k) = k$ and $\mathrm{ran}(\mathbf{U}_{\mathrm{E},k}) = \mathrm{ran}(\mathbf{E}_k)$. Therefore, $\tilde{\mathbf{D}}$ is the component of $\mathbf{D}$ orthogonal to the column space of $\mathbf{E}_k$. As a consequence, $\tilde{\mathbf{D}}$ is referred to be the orthogonal component of $\mathbf{D}$. The thin QR-decomposition of $\tilde{\mathbf{D}}$ is expressed as $\tilde{\mathbf{D}} = \mathbf{Q}\mathbf{R}$, where $\mathbf{Q} \in \mathbb{R}^{m\times p}$ and $\mathbf{R} \in \mathbb{R}^{p\times p}$.

Define $\hat{\mathbf{B}} = [\mathbf{E}_k, \mathbf{D}]$, which can be expressed as,

$$\hat{\mathbf{B}} = \begin{bmatrix}\mathbf{U}_{\mathrm{E},k} & \mathbf{Q}\end{bmatrix}\mathbf{M}\begin{bmatrix}\mathbf{V}_{\mathrm{E},k}^{\mathrm{T}} & \mathbf{0}\\ \mathbf{0} & \mathbf{I},\end{bmatrix}$$

where $\mathbf{M} = \begin{bmatrix}\mathbf{S}_{\mathrm{E},k} & \mathbf{U}_{\mathrm{E},k}^{\mathrm{T}}\mathbf{D}\\ \mathbf{0} & \mathbf{R}\end{bmatrix}$ and $\mathbf{I}$ is the identity matrix of appropriate size. Denote the SVD of $\mathbf{M}$ as $\mathbf{M} = \mathbf{U}_{\mathrm{M}}\mathbf{S}_{\mathrm{M}}\mathbf{V}_{\mathrm{M}}^{\mathrm{T}}$.

Consequently, the rank-k SVD of $\hat{\mathbf{B}}$ is given by $\hat{\mathbf{B}}_k = \mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}\mathbf{V}_{\hat{\mathrm{B}},k}^{\mathrm{T}}$ [16], where

$$\begin{aligned}\mathbf{U}_{\hat{\mathrm{B}},k} &= \begin{bmatrix}\mathbf{U}_{\mathrm{E},k} & \mathbf{Q}\end{bmatrix}\mathbf{U}_{\mathrm{M},k}, \quad \mathbf{S}_{\hat{\mathrm{B}},k} = \mathbf{S}_{\mathrm{M},k},\\ \mathbf{V}_{\hat{\mathrm{B}},k} &= \begin{bmatrix}\mathbf{V}_{\mathrm{E},k} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\mathbf{V}_{\mathrm{M},k}.\end{aligned} \tag{2}$$

D. MOTIVATION

Notice that $\mathbf{U}_{\hat{\mathrm{B}},k}$ and $\mathbf{V}_{\hat{\mathrm{B}},k}$ are the exact dominant singular matrices of $\hat{\mathbf{B}}$. In [20], it is shown that if

$$\begin{aligned}&(\mathbf{E}_k^{\mathrm{c}})^{\mathrm{T}}\mathbf{D} = \mathbf{0}\\ &\text{and } \sigma_k(\hat{\mathbf{B}}) > \sigma_{k+1}(\mathbf{B}),\end{aligned} \tag{3}$$

then $\hat{\mathbf{B}}_k = \mathbf{B}_k$. That is, the best rank-k approximation of $\hat{\mathbf{B}}$ is exactly that of $\mathbf{B}$. In this case, there is no loss in approximating $\mathbf{E}$ by $\mathbf{E}_k$ in the updating procedure. The first condition in (3) indicates that, for $\hat{\mathbf{B}}_k = \mathbf{B}_k$ to occur, the new columns in $\mathbf{D}$ should be orthogonal to the discarded component $\mathbf{E}_k^{\mathrm{c}}$. For example, if the columns of $\mathbf{D}$ lie in the column space of $\mathbf{E}_k$, one has $(\mathbf{E}_k^{\mathrm{c}})^{\mathrm{T}}\mathbf{D} = \mathbf{0}$. A class of matrices which satisfies (3) is the low-rank-plus-shift matrices [16]. Matrices with this structure can be represented as $\mathbf{B}^{\mathrm{T}}\mathbf{B} = \mathbf{C} + \alpha^2\mathbf{I}$, where $\mathbf{C}$ is positive semidefinite with $\mathrm{rank}(\mathbf{C}) = k$.

However, since the new columns in $\mathbf{D}$ may lie in arbitrary subspaces of $\mathbb{R}^m$, conditions (3) cannot be satisfied in most circumstances. Especially, when a change point occurs, the column space of $\mathbf{D}$ may change severely from that of $\mathbf{E}_k$. In this case, the above updating procedure will lead to an inaccurate matrix decomposition. As a consequence, $\mathbf{U}_{\hat{\mathrm{B}},k}$ and $\mathbf{V}_{\hat{\mathrm{B}},k}$ are only approximations of the dominant singular matrices of $\mathbf{B}$.

Generally, suppose $\tilde{\mathbf{D}}$ has full column rank. Then the orthonormal columns of $\mathbf{Q}$ span the column space of $\tilde{\mathbf{D}}$. Therefore, in this work, we say that $\mathbf{Q}$ contains the column information of the orthogonal component $\tilde{\mathbf{D}}$. In equation (2), it is shown that $\mathbf{Q}$ is utilized in calculating $\mathbf{U}_{\hat{\mathrm{B}},k}$. But in the calculation of $\mathbf{V}_{\hat{\mathrm{B}},k}$, this information is not utilized.

However, in some applications, it is important to capture the column information of $\tilde{\mathbf{D}}$ in the right singular space. For example, in LSI, each document is represented by the corresponding length-k vector in the right singular matrix $\mathbf{V}_{\hat{\mathrm{B}},k}$ [18]. However, when the column space of the added columns $\mathbf{D}$ changes significantly from $\mathrm{ran}(\mathbf{E}_k)$, the major component of $\mathbf{D}$ is contained in $\tilde{\mathbf{D}}$. In this case, due to the loss of the column information of $\tilde{\mathbf{D}}$ in $\mathbf{V}_{\hat{\mathrm{B}},k}$, the newly added documents (columns) may not be reasonably characterized in the updated right singular space. Consequently, one apparent source of the inaccuracy in the updating of $\mathbf{V}_{\hat{\mathrm{B}},k}$ is that it captures less information of the orthogonal component $\tilde{\mathbf{D}}$. Worse still, the resulting error will accumulate over the SVD updating iterations. This will be verified by simulation results in Section V-B.

Based on incSVD, this work proposes an EincSVD algorithm. In EincSVD, more information of the orthogonal component $\tilde{\mathbf{D}}$ is utilized in updating the right singular space. It will be shown that the approximation error of EincSVD is smaller than that of incSVD. Furthermore, EincSVD performs much better than incSVD in detecting the change points of dynamic networks.

III. PROPOSED ALGORITHMS

In this section, the proposed EincSVD algorithm is provided. Moveover, theoretical analysis on the approximation error demonstrates that EincSVD is a more accurate decomposition than incSVD. To reduce the computational complexity, an AEincSVD algorithm is proposed as an adaptive combination of EincSVD and incSVD.

A. ENHANCED INCREMENTAL SVD (EINCSVD)

In order to overcome the above mentioned problem, an enhanced version of incSVD is proposed. EincSVD differs from incSVD only in the calculation of the right singular matrix. In EincSVD, the modified right singular matrix is computed as

$$\mathbf{V}'_{\hat{\mathrm{B}},k} = \mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}'^{-1}_{\mathrm{B},k}, \tag{4}$$

where $\mathbf{S}'^{-1}_{\mathrm{B},k}$ is a diagonal matrix used to normalize the columns of $\mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}$.

Consider the range of $\mathbf{V}'_{\hat{\mathrm{B}},k}$,

$$\begin{aligned}&\mathrm{ran}(\mathbf{V}'_{\hat{\mathrm{B}},k})\\ &= \mathrm{ran}(\mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k})\\ &= \mathrm{ran}\left(\begin{bmatrix}\mathbf{E}^{\mathrm{T}}\\ \mathbf{D}^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}\mathbf{U}_{\mathrm{E},k} & \mathbf{Q}\end{bmatrix}\mathbf{U}_{\mathrm{M},k}\right)\\ &= \mathrm{ran}\left(\begin{bmatrix}\mathbf{E}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k}\mathbf{S}_{\mathrm{E},k} & \mathbf{E}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k}\mathbf{U}_{\mathrm{E},k}^{\mathrm{T}}\mathbf{D} + \mathbf{E}^{\mathrm{T}}\tilde{\mathbf{D}}\\ \mathbf{D}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k}\mathbf{S}_{\mathrm{E},k} & \mathbf{D}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k}\mathbf{U}_{\mathrm{E},k}^{\mathrm{T}}\mathbf{D} + \mathbf{D}^{\mathrm{T}}\tilde{\mathbf{D}}\end{bmatrix}\mathbf{V}_{\mathrm{M},k}\right).\end{aligned} \tag{5}$$

The third equation uses the fact that $\mathbf{U}_{\mathrm{M},k} = \mathbf{M}_k\mathbf{V}_{\mathrm{M},k}\mathbf{S}_{\mathrm{M},k}^{-1} = \mathbf{M}\mathbf{V}_{\mathrm{M},k}\mathbf{S}_{\mathrm{M},k}^{-1}$.

Recall that the right singular matrix $\mathbf{V}_{\hat{\mathrm{B}},k}$ in incSVD can be expressed as,

$$\mathbf{V}_{\hat{\mathrm{B}},k} = \hat{\mathbf{B}}_k^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}^{-1}. \tag{6}$$

Using $\mathbf{E}_k^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k} = \mathbf{E}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k}$, $\hat{\mathbf{B}}_k^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k} = \hat{\mathbf{B}}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}$ and $\mathbf{E}_k^{\mathrm{T}}\tilde{\mathbf{D}} = \mathbf{0}$, one has

$$\begin{aligned}&\mathrm{ran}(\mathbf{V}_{\hat{\mathrm{B}},k})\\ &= \mathrm{ran}(\hat{\mathbf{B}}_k^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k})\\ &= \mathrm{ran}\left(\begin{bmatrix}\mathbf{E}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k}\mathbf{S}_{\mathrm{E},k} & \mathbf{E}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k}\mathbf{U}_{\mathrm{E},k}^{\mathrm{T}}\mathbf{D}\\ \mathbf{D}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k}\mathbf{S}_{\mathrm{E},k} & \mathbf{D}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k}\mathbf{U}_{\mathrm{E},k}^{\mathrm{T}}\mathbf{D} + \mathbf{D}^{\mathrm{T}}\tilde{\mathbf{D}}\end{bmatrix}\mathbf{V}_{\mathrm{M},k}\right).\end{aligned} \tag{7}$$

Comparing (5) and (7), one observes that there is an additional $\mathbf{E}^{\mathrm{T}}\tilde{\mathbf{D}}$ in the upper right submatrix in (5). Therefore, EincSVD is different from incSVD in that more information of $\tilde{\mathbf{D}}$ is utilized in calculating the right singular space.

When the graph stream is updated with a new graph segment being added, the SVD of the new edge-segment matrix can be obtained through incremental SVD algorithms. Analogous to LSI [18], the graph segments and edges are characterized by the row vectors in the right and left singular matrices, respectively. The interpretation of the proposed EincSVD algorithm (4) is that, each graph segment is represented as a combination of its constituent edge vectors, weighted by the real edge-segment matrix $\mathbf{B}$ rather than $\hat{\mathbf{B}}_k$ in (6). Intuitively, $\mathbf{U}_{\hat{\mathrm{B}},k}$ and $\mathbf{V}'_{\hat{\mathrm{B}},k}$ will be a more accurate decomposition of $\mathbf{B}$. This will be verified in Theorem 1.

Define $\hat{\mathbf{B}}'_k = \mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}\mathbf{V}'^{\mathrm{T}}_{\hat{\mathrm{B}},k}$, and one has the following theorem.

Theorem 1: Compared with $\hat{\mathbf{B}}_k$, $\hat{\mathbf{B}}'_k$ is a better approximation to the original matrix $\mathbf{B}$ in the Frobenius norm. That is, $||\mathbf{B}-\hat{\mathbf{B}}'_k||^2_{\mathrm{F}} \leq ||\mathbf{B}-\hat{\mathbf{B}}_k||^2_{\mathrm{F}}$.

Proof: To prove Theorem 1 is equivalent to prove

$$\mathrm{tr}\left((\mathbf{B}-\hat{\mathbf{B}}'_k)^{\mathrm{T}}(\mathbf{B}-\hat{\mathbf{B}}'_k)\right) \leq \mathrm{tr}\left((\mathbf{B}-\hat{\mathbf{B}}_k)^{\mathrm{T}}(\mathbf{B}-\hat{\mathbf{B}}_k)\right), \quad (8)$$

where $\mathrm{tr}(\cdot)$ denotes the trace of a matrix.

Equation (8) is equivalent to

$$-2\mathrm{tr}(\mathbf{B}^{\mathrm{T}}\hat{\mathbf{B}}'_k) + \mathrm{tr}(\hat{\mathbf{B}}'^{\mathrm{T}}_k\hat{\mathbf{B}}'_k) \leq -2\mathrm{tr}(\mathbf{B}^{\mathrm{T}}\hat{\mathbf{B}}_k) + \mathrm{tr}(\hat{\mathbf{B}}^{\mathrm{T}}_k\hat{\mathbf{B}}_k). \quad (9)$$

Notice that

$$\begin{aligned}\mathrm{tr}(\hat{\mathbf{B}}'^{\mathrm{T}}_k\hat{\mathbf{B}}'_k) &= \mathrm{tr}(\mathbf{V}'_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}\mathbf{U}^{\mathrm{T}}_{\hat{\mathrm{B}},k}\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}\mathbf{V}'^{\mathrm{T}}_{\hat{\mathrm{B}},k}) \\ &= \mathrm{tr}(\mathbf{S}^2_{\hat{\mathrm{B}},k}), \quad (10)\\ \mathrm{tr}(\hat{\mathbf{B}}^{\mathrm{T}}_k\hat{\mathbf{B}}_k) &= \mathrm{tr}(\mathbf{V}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}\mathbf{U}^{\mathrm{T}}_{\hat{\mathrm{B}},k}\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}\mathbf{V}^{\mathrm{T}}_{\hat{\mathrm{B}},k}) \\ &= \mathrm{tr}(\mathbf{S}^2_{\hat{\mathrm{B}},k}). \quad (11)\end{aligned}$$

Substituting (10) and (11) into (9), one has $\mathrm{tr}(\mathbf{B}^{\mathrm{T}}\hat{\mathbf{B}}'_k) \geq \mathrm{tr}(\mathbf{B}^{\mathrm{T}}\hat{\mathbf{B}}_k)$, which can be written as

$$\mathrm{tr}(\mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}\mathbf{V}'^{\mathrm{T}}_{\hat{\mathrm{B}},k}) \geq \mathrm{tr}(\mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}\mathbf{V}^{\mathrm{T}}_{\hat{\mathrm{B}},k}). \quad (12)$$

Equation (12) can be further expressed as,

$$\mathrm{tr}(\mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}\mathbf{S}'^{-1}_{\mathrm{B},k}\mathbf{U}^{\mathrm{T}}_{\hat{\mathrm{B}},k}\mathbf{B}) \geq \mathrm{tr}(\mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}\mathbf{S}^{-1}_{\hat{\mathrm{B}},k}\mathbf{U}^{\mathrm{T}}_{\hat{\mathrm{B}},k}\hat{\mathbf{B}}_k). \quad (13)$$

Define $\mathbf{X} = \mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}_{\hat{\mathrm{B}},k}$, $\mathbf{Y}^* = \mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}'^{-1}_{\mathrm{B},k}$ and $\mathbf{Z} = \hat{\mathbf{B}}^{\mathrm{T}}_k\mathbf{U}_{\hat{\mathrm{B}},k}\mathbf{S}^{-1}_{\hat{\mathrm{B}},k}$. Equation (13) can be rewritten as

$$\mathrm{tr}(\mathbf{X}\mathbf{Y}^{*\mathrm{T}}) \geq \mathrm{tr}(\mathbf{X}\mathbf{Z}^{\mathrm{T}}),$$

or equivalently,

$$\mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{Y}^*) \geq \mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{Z}). \quad (14)$$

Let $\mathcal{F}_{\mathrm{X}}$ denote the set of matrices, which are of the same size as $\mathbf{X}$ and have normalized columns. Apparently, $\mathbf{Y}^*, \mathbf{Z} \in \mathcal{F}_{\mathrm{X}}$, and $\mathbf{y}^*_i = \mathbf{x}_i/|\mathbf{x}_i|$.

In the following, it will be proved that $\mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{Y}^*) \geq \mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{Y})$ holds for all $\mathbf{Y} \in \mathcal{F}_{\mathbf{X}}$.

Consider the optimization problem

$$\begin{aligned}&\underset{\mathbf{Y}}{\text{maximize}}\ \mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{Y})\\ &\text{subject to } \mathbf{Y} \in \mathcal{F}_{\mathrm{X}}.\end{aligned}$$

The object function can be written as $\mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{Y}) = \sum_{i=1}^{k}\mathbf{x}^{\mathrm{T}}_i\mathbf{y}_i$. From Cauchy-Schwarz inequality [21], it is known that for each item $\max_{|\mathbf{y}_i|=1}\mathbf{x}^{\mathrm{T}}_i\mathbf{y}_i$, the optimal point is $\mathbf{y}^*_i = \mathbf{x}_i/|\mathbf{x}_i|$. As a consequence, $\mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{Y}^*) \geq \mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{Y})$ satisfies for all matrix $\mathbf{Y} \in \mathcal{F}_{\mathbf{X}}$.

This completes the proof. ■

Theorem 1 shows that through an alternative way to calculate the right singular matrix, EincSVD provides a better decomposition of the updated matrix.

B. ADAPTIVE EINCSVD (AEINCSVD)

As proposed in (4), the EincSVD algorithm involves a matrix multiplication with matrix $\mathbf{B}$. In general, EincSVD is more complex than the less accurate alternative incSVD. In this subsection, AEincSVD is proposed as an adaptive combination of incSVD and EincSVD to reduce the complexity.

According to the discussions in Section II-D, if the column space ran($\mathbf{D}$) does not change much from ran($\mathbf{U}_{\mathrm{E},k}$), one has $\tilde{\mathbf{D}} \approx \mathbf{0}$. In this case, the loss of the column information of $\tilde{\mathbf{D}}$ in $\mathbf{V}_{\hat{\mathrm{B}},k}$ is negligible. The approximation error of incSVD is more obvious when ran($\mathbf{D}$) changes significantly from ran($\mathbf{U}_{\mathrm{E},k}$) = ran($\mathbf{E}_k$). Therefore, in order to reduce the updating complexity, the deviation of ran($\mathbf{D}$) from ran($\mathbf{U}_{\mathrm{E},k}$) can be monitored to determine when it is necessary to switch from incSVD to EincSVD. In the updating procedure, EincSVD is invoked only when necessary.

To this end, one can monitor the cosine similarities $s_{i,j}$ between the added columns $\mathbf{d}_i$ ($1 \leq i \leq p$) and the columns of $\mathbf{U}_{\mathrm{E},k} = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_k]$,

$$s_{i,j} = \frac{\mathbf{d}^{\mathrm{T}}_i\mathbf{u}_j}{|\mathbf{d}_i||\mathbf{u}_j|} = \frac{\mathbf{d}^{\mathrm{T}}_i}{|\mathbf{d}_i|}\mathbf{u}_j, \quad 1 \leq i \leq p,\ 1 \leq j \leq k.$$

The deviation of $\mathbf{d}_i$ from ran($\mathbf{U}_{\mathrm{E},k}$) is characterized by d_i [22], where

$$d_i = 1 - \sum_{j=1}^{k} s^2_{i,j}.$$

The cosine similarity $s_{i,j}$ is the projection of unit vector $\mathbf{d}^{\mathrm{T}}_i/|\mathbf{d}_i|$ onto unit vector $\mathbf{u}_j$. Since $\mathbf{U}_{\mathrm{E},k}$ forms an orthonormal basis in Euclidean space, the squares of these projections can at most sum to one. Therefore, it is easy to verify that the range of d_i is $[0, 1]$. If vector $\mathbf{d}_i$ lies in the column space of $\mathbf{U}_{\mathrm{E},k}$, then one has $d_i = 0$. At the other extreme, one has $d_i = 1$ if $\mathbf{d}_i$ is orthogonal to each column in $\mathbf{U}_{\mathrm{E},k}$. Define $d = \max_i\{d_i\}$ to denote the deviation of ran($\mathbf{D}$) from ran($\mathbf{U}_{\mathrm{E},k}$). In the AEincSVD Algorithm, EincSVD is invoked when d is larger than some empirical threshold.

AEincSVD offers an improvement in complexity when compared with EincSVD or recomputing the batch SVD. The complexity of AEincSVD is only slightly higher than

that of incSVD. At the same time, as will be shown in Section V, AEincSVD can provide much better performance than incSVD in terms of approximation error and in detecting the change points of dynamic networks.

IV. COMPUTATIONAL COMPLEXITY ANALYSIS

In this section, the computational cost of incSVD [16], the proposed EincSVD, and the batch SVD is compared. It will be shown that EincSVD is more efficient than the batch SVD algorithm.

For the complexity of incSVD, calculating $\tilde{\mathbf{D}}$ requires $\mathcal{O}(mkp)$ operations. The complexity of performing a QR decomposition on $\tilde{\mathbf{D}}$ is $\mathcal{O}(mp^2)$. The SVD of matrix $\mathbf{M}$ costs $\mathcal{O}\left((k+p)^3\right)$. At last, computing $\mathbf{U}_{\hat{\mathrm{B}},k}$ and $\mathbf{V}_{\hat{\mathrm{B}},k}$ requires complexity $\mathcal{O}\left(mk(k+p)\right)$ and $\mathcal{O}(nk^2)$, respectively.

EincSVD differs from incSVD only in calculating the right singular matrix. It is generally more complex than the less accurate incSVD algorithm. Equation (4) involves a matrix multiplication with complexity $\mathcal{O}(mnk)$. On the other hand, $\mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k}$ can be expressed as,

$$\mathbf{B}^{\mathrm{T}}\mathbf{U}_{\hat{\mathrm{B}},k} = \begin{bmatrix} \mathbf{E}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k} & \mathbf{E}^{\mathrm{T}}\mathbf{Q} \\ \mathbf{D}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k} & \mathbf{D}^{\mathrm{T}}\mathbf{Q} \end{bmatrix} \mathbf{U}_k. \tag{15}$$

If $\mathbf{E}^{\mathrm{T}}\mathbf{U}_{\mathrm{E},k}$ is already calculated in the previous iteration, the complexity of computing (15) is $\mathcal{O}(mnp)$. Therefore, computing $\mathbf{V}'_{\hat{\mathrm{B}},k}$ in the proposed algorithm requires $\mathcal{O}\left(mn\ \min\{k,p\}\right)$ operations. Notice that most of the computational cost of the proposed algorithm is spent in the matrix multiplication of (4). If only one column is added in each step, the complexity is reduced to $\mathcal{O}(mn)$. In the case that $\mathbf{B}$ is a binary matrix or $\mathbf{B}$ is sparse, the computational cost can be tremendously reduced.

The standard procedures for computing the batch SVD have a computational complexity of $\mathcal{O}(n^2\ m)$ (with $m \leq n$) [19]. For the rank-k SVD, Lanczos method [23] is often used. The Lanczos method consists of a bidiagonalization step, a step of SVD on the bidiagonal matrix, and a Ritz transformation step. The most expensive step is the first one, which requires iterations of matrix-vector multiplications with matrix $\mathbf{B}$ and the complexity is at least $\mathcal{O}(mnk)$.

Focus on the most expensive steps in the proposed EincSVD algorithm and Lanczos method. The matrix multiplication (4) in EincSVD requires at most $m(n+p)k$ scalar multiplications. However, although the bidiagonalization in Lanczos method can be computed with a complexity on the order of at least $\mathcal{O}(mnk)$, a substantial number of matrix-vector multiplications is required until the algorithm converges. The cost of bidiagonalization is in fact $Km(n+p)k$ with $K \gg 1$. Therefore, although the proposed incremental algorithm EincSVD is more complex than incSVD, it is still much more efficient than computing the SVD from scratch. This can also be demonstrated by comparisons of computational time in Section V-D.

V. PERFORMANCE EVALUATION

In this section, we present performance evaluations of the proposed algorithms. Firstly, a synthetic dataset demonstrating a dynamic network is introduced. Then comparisons of different algorithms in terms of approximation error, in detecting change points and in computational complexity are sequentially provided.

A. SYNTHETIC DATASET

In the simulation, the performance of the proposed algorithms is evaluated with a synthetic network model similar to [13], which is an extended dynamic stochastic block model [24]. A graph stream including 80 random graphs are generated. In each graph snapshot, a total of 64 nodes is partitioned into 4 equal-size communities of 16 nodes. Three synthetic change points separate the stream into 4 fragments: $G^{(1)} \sim G^{(20)}$, $G^{(21)} \sim G^{(40)}$, $G^{(41)} \sim G^{(60)}$ and $G^{(61)} \sim G^{(80)}$. The community structure is shown in Table 1.

Suppose each node has on average 6 edges within the community and 2 edges to members of other communities. The weight of an edge is drawn uniformly from 1 to 10 for intra-community edges, while from 1 to 6 for inter-community edges.

In the graph stream, each graph snapshot comes in one after the other. A graph segment is assumed to consist of one snapshot. As a start point, it is assumed that one has a rank-k batch SVD of the edge-segment matrix $\mathbf{E}$ at $t_0 = 10$. Then one column is added to $\mathbf{E}$ at each step. Incremental algorithms incSVD and EincSVD are applied iteratively to update the SVD of $\mathbf{E}$. In the following, it is assumed that $k = 3$ unless specified otherwise.

B. APPROXIMATION ERROR OF INCREMENTAL ALGORITHMS

The relative error of the best rank-k approximation $\mathbf{E}_k$ is defined as,

$$e_{\mathrm{opt},k} = \frac{||\mathbf{E} - \mathbf{E}_k||_{\mathrm{F}}^2}{||\mathbf{E}||_{\mathrm{F}}^2}. \tag{16}$$

In the following, the subscript k will be omitted when it is clear from the context. Relative errors e_{inc}, e_{Einc}, and e_{AEinc} can be defined similarly if incSVD, EincSVD and AEincSVD are used in updating the SVD.

The difference between the relative errors of incremental SVD algorithms and that of the best rank-k approximation is depicted in Fig. 1. Theorem 1 is verified by the result since the difference between EincSVD and the best rank-k approximation is always smaller than that of incSVD. Notice that after the change points at $t = 20$ and $t = 40$, there are obvious increases in the approximation error for incremental algorithms. The performance of the proposed EincSVD is much better than that of incSVD. Fig. 1 shows that EincSVD gradually reduces the difference for some extent. But the difference between incSVD and the best approximation tends to diverge.

TABLE 1. The community structure of the dynamic graphs.

Graphs	Community 1	Community 2	Community 3	Community 4
$G^{(1)} \sim G^{(20)}$	{1∼16}	{17∼32}	{33∼48}	{49∼64}
$G^{(21)} \sim G^{(40)}$	{1∼12}∪{61∼64}	{13∼28}	{29∼44}	{45∼60}
$G^{(41)} \sim G^{(60)}$	{1∼8}∪{57∼64}	{9∼24}	{25∼40}	{41∼56}
$G^{(61)} \sim G^{(80)}$	{1∼4}∪{53∼64}	{5∼20}	{21∼36}	{37∼52}

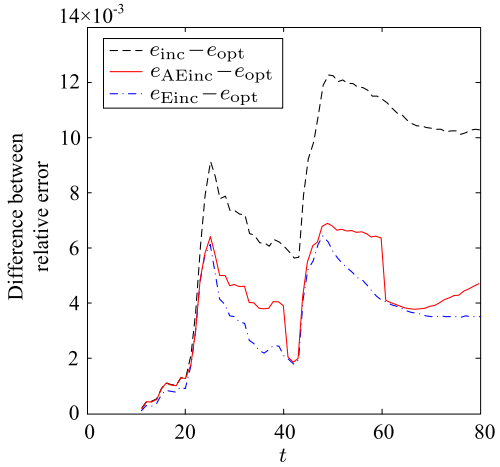


FIGURE 1. Difference between relative errors of different SVD algorithms.

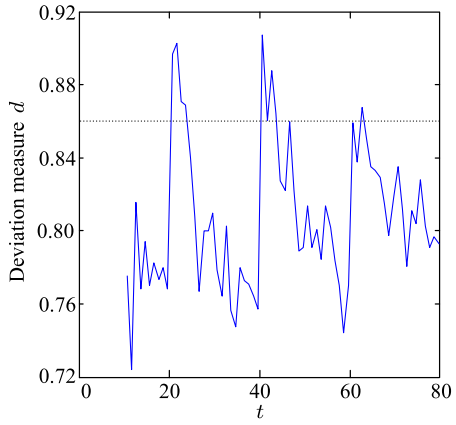


FIGURE 2. Deviation d of each graph snapshot from previous ones.

In Section III-B, AEincSVD is proposed as a combination of incSVD and EincSVD. In the updating procedure, incSVD is performed if the direction of the new columns does not change much from previous columns. When the derivation of newly added columns from existing ones exceeds a threshold, the algorithm switches to EincSVD to recover the column information of $\tilde{\mathbf{D}}$.

The deviation d of each column from previous ones is plotted in Fig. 2. The curve shows three overshoots at the change points, indicating that the direction of newly added columns deviates severely from existing ones. According to Fig. 2, it is assumed that incSVD is switched to EincSVD if $d > 0.86$ in the AEincSVD algorithm. Therefore, the complexity of AEincSVD is much lower than that of EincSVD. The difference between the relative error of AEincSVD and that of the best rank-k approximation is also depicted in Fig. 1. As shown in Fig. 1, the relative error of AEincSVD is close to that of EincSVD and much smaller than that of incSVD.

C. INCREMENTAL SVD ALGORITHMS FOR CHANGE POINT DETECTION

In this subsection, the performance of the proposed algorithms in detecting change points is evaluated.

Fig. 3 and Fig. 4 depict the cosine distance between consecutive graph segments at time $t = 60$ and $t = 61$, respectively. In the interval of $t = 1 \sim 60$, there are two change points at $t = 20$ and $t = 40$, which are successfully detected by the proposed EincSVD algorithm (see Fig. 3(b)). However, Fig. 3(a) shows that incSVD does not clearly identify the two change points. The advantage of EincSVD over incSVD is obviously shown.

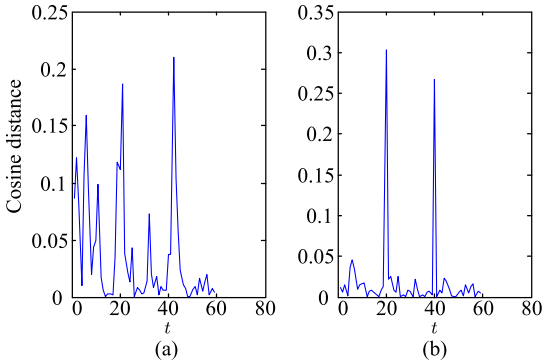


FIGURE 3. Change point detection results at $t = 60$ of (a) incSVD; (b) EincSVD.

An additional change point occurs between $t = 60$ and $t = 61$. As illustrated in Fig. 4, the two incremental algorithms clearly identify the third change point at time $t = 61$. Comparison between Fig. 3 and Fig. 4 indicates that by updating the SVD through incremental algorithms, the changes of the dynamic network can be detected in real time, rather than waiting to collect all the graph data. In addition, the updating method avoids recomputing the SVD from scratch when new graphs are added to the data collection.

In the following, the algorithms will be compared in more detail. Unless specified otherwise, the detection results at $t = 80$ are considered.

The results of change point detection at $t = 80$ of rank-k batch SVD, incSVD, EincSVD, and AEincSVD are

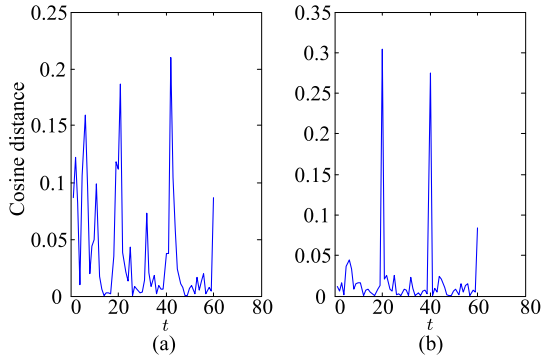


FIGURE 4. Change point detection results at $t = 61$ of (a) incSVD; (b) EincSVD.

illustrated in Fig. 5. It is shown that the three change points are clearly and correctly identified by the rank-k batch SVD, the proposed EincSVD and AEincSVD algorithms. However, similar to Fig. 3(a) and Fig. 4(a), Fig. 5(b) shows that the incSVD algorithm only detects a real change point at $t = 60$ and returns lots of false alarms. This indicates that incSVD is not appropriate for the use of change point detection. According to (1), the detection is mainly based on the right singular space. Through an alternative approach to calculate the right singular matrix, the proposed algorithms perform much better than incSVD in detecting the change points.

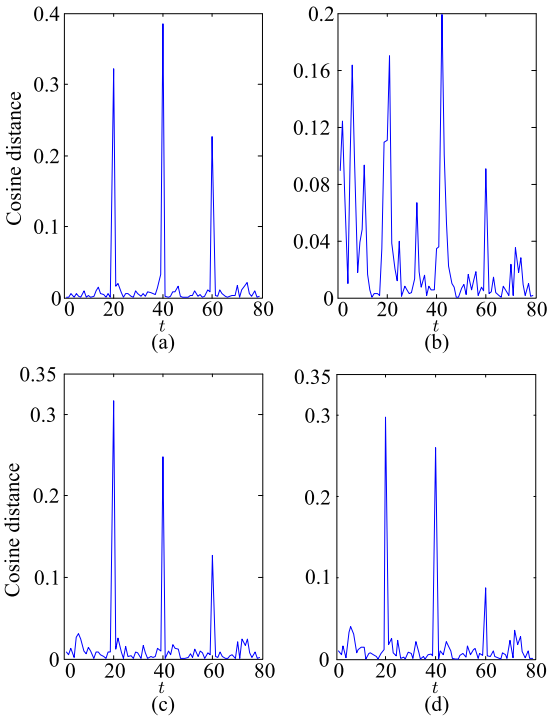


FIGURE 5. Change point detection results at $t = 80$ of (a) Batch SVD; (b) incSVD; (c) EincSVD; (d) AEincSVD.

To proceed, the precision-recall performance of different algorithms is examined. The simulation consists of 100 random realizations of the above described dynamic network.

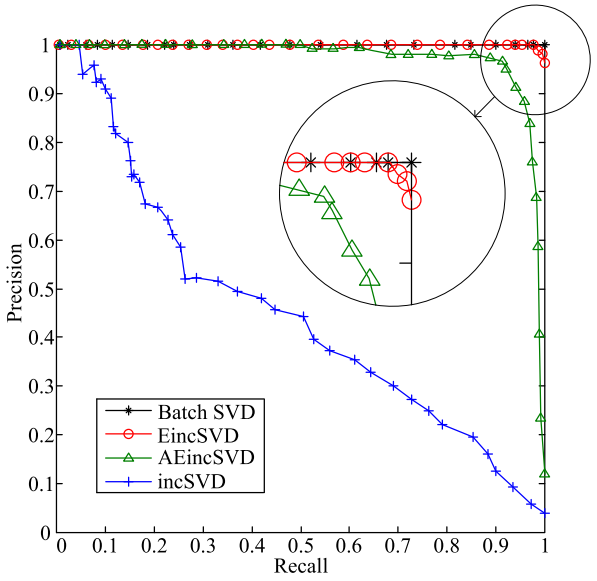


FIGURE 6. Precision-recall curves of different algorithms.

Fig. 6 depicts the precision-recall curves of the change point detection results at time $t = 80$. Precision is calculated as the ratio of correctly detected change points relative to the total number of points detected, and recall as the ratio of correctly detected change points relative to the total number of change points. As illustrated in Fig. 6, the precision of the rank-k batch SVD remains perfect at 100% for all levels of recall in our simulation. This demonstrates that the LSI-based approach can effectively detect changes of the community structure in dynamic networks. The performance of EincSVD is extremely close the batch SVD. It is shown that 98% of the change points are identified by EincSVD with 100% precision. However, the incSVD performs much worse. With incSVD, less than 5% of the change points are detected with full precision. The precision level drops rapidly to below 0.1 for high recall values. As analyzed in Section II-D, this is mainly due to the loss of the colulmn information of $\tilde{\mathbf{D}}$ in the approximated right singular space, and the accumulation of this error in the iterative updating procedure. Following the results in Section V-B, the threshold of deviation measure in the AEincSVD algorithm is set to be 0.86. As a combination of incSVD and EincSVD, the complexity of AEincSVD is slightly higher than that of incSVD, and it provides excellent detection performance comparable to EincSVD (see Fig. 6).

D. COMPARISON OF COMPUTATIONAL COMPLEXITY

In the following, the computational complexity of different SVD algorithms is compared. The complexity is tested on the MED dataset [25]. The MED matrix consists of 5831 rows and 1033 columns with 52003 nonzero elements. For the computation time of the rank-k batch SVD, the minimum among the running times of the thin SVD, the Lanczos SVD [26] and the partial SVD (by calling the Matlab built-in function svds) is returned. For incSVD and EincSVD,

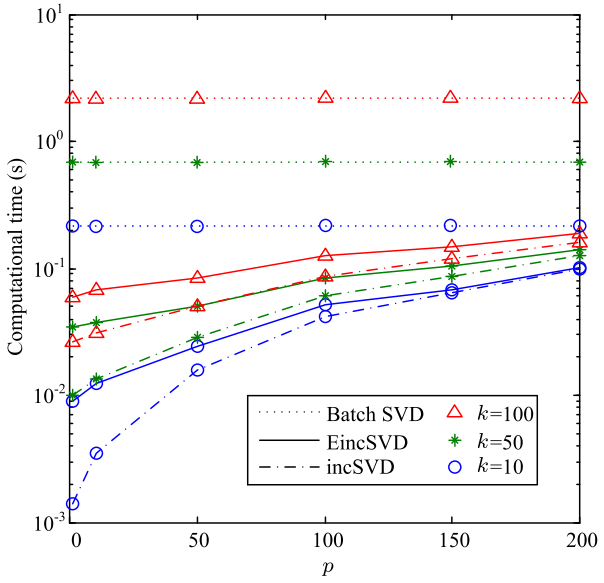


FIGURE 7. Computational complexity of different algorithms.

the results are presented for different k and numbers of added columns p. For example, with $p = 100$, the computation time corresponds to updating the rank-k SVD from $\mathbf{E} \in \mathbb{R}^{5831\times 933}$ to $\mathbf{B} = [\mathbf{E}, \mathbf{D}] \in \mathbb{R}^{5831\times 1033}$ by using incSVD or EincSVD.

As illustrated in Fig. 7, the complexity of incremental SVD algorithms increases with k and p. The computation time of the batch algorithm increases only with k since it is irrelevant to p. It is shown that although the proposed EincSVD is more complex than incSVD, it requires much less complexity than computing the batch SVD from scratch. With $k = 100$ and various p, EincSVD is more efficient than the batch SVD by more than one order of magnitude. When p is small, for example, if only one column is added at each time, the improvement of EincSVD over batch SVD in complexity is especially obvious. These results validate the efficiency of using incremental algorithms in updating the SVD of large matrices.

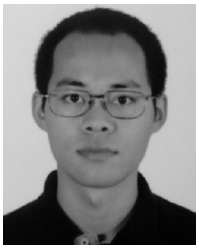
VI. CONCLUSIONS

In order to dynamically detect the change points of network structures, an enhanced incremental SVD algorithm EincSVD is proposed in this work. In the EincSVD algorithm, more information of the added columns is utilized in updating the right singular space. It is proved that the approximation error of EincSVD is always smaller than that of the traditional incSVD algorithm. Although EincSVD requires more computational cost than incSVD, it is still much more efficient than recomputing the batch SVD from scratch. To further reduce the complexity, another AEincSVD algorithm is proposed as an adaptive combination of incSVD and EincSVD. Simulations on detecting the change points in dynamic networks are conducted. It is shown that the proposed incremental algorithms are effective in detecting the changes of the community structure. In addition, the performance of EincSVD is very close to that of the batch SVD approach. Both EincSVD and AEincSVD perform much better than the incSVD algorithm.

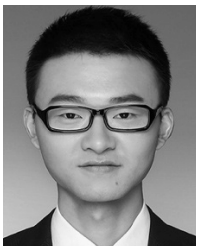
REFERENCES

[1] T. Idé and H. Kashima, "Eigenspace-based anomaly detection in computer systems," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2004, pp. 440–449.
[2] S. Ranshous, S. Shen, D. Koutra, S. Harenberg, C. Faloutsos, and N. F. Samatova, "Anomaly detection in dynamic networks: A survey," *Wiley Interdiscipl. Rev., Comput. Stat.*, vol. 7, no. 3, pp. 223–247, 2015.
[3] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Mining Knowl. Discovery*, vol. 29, no. 3, pp. 626–688, 2015.
[4] I. Balabine and A. Velednitsky, "Method and system for confident anomaly detection in computer network traffic," U.S Patent 9 843 488 B2, Dec. 12, 2017.
[5] C. Phua, V. Lee, K. Smith, and R. Gayler. (2010). "A comprehensive survey of data mining-based fraud detection research." [Online]. Available: https://arxiv.org/abs/1009.6119
[6] A. Y. Mutlu and S. Aviyente, "Dynamic network summarization using convex optimization," in *Proc. IEEE Stat. Signal Process. Workshop (SSP)*, Aug. 2012, pp. 117–120.
[7] S. Koujaku, M. Kudo, I. Takigawa, and H. Imai, "Structual change point detection for evolutional networks," in *Proc. World Congr. Eng.*, vol. 1, 2013, pp. 324–329.
[8] C. E. Priebe, J. M. Conroy, Y. Park, and D. J. Marchette, "Scan statistics on enron graphs," *Comput. Math. Org. Theory*, vol. 11, no. 3, pp. 229–247, 2005.
[9] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu, "Graphscope: Parameter-free mining of large time-evolving graphs," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2007, pp. 687–696.
[10] P. Wills and F. G. Meyer. (2017). "Detecting topological changes in dynamic community networks." [Online]. Available: https://arxiv.org/abs/1707.07362
[11] N. D. Monnig and F. G. Meyer, "The resistance perturbation distance: A metric for the analysis of dynamic networks," *Discrete Appl. Math.*, vol. 236, pp. 347–386, Feb. 2018.
[12] Y. Yasami and F. Safaei, "A statistical infinite feature cascade-based approach to anomaly detection for dynamic social networks," *Comput. Commun.*, vol. 100, pp. 52–64, Mar. 2017.
[13] D. Duan, Y. Li, Y. Jin, and Z. Lu, "Community mining on dynamic weighted directed graphs," in *Proc. 1st ACM Int. Workshop Complex Netw. Meet Inf. Knowl. Manage. (CNIKM)*, New York, NY, USA, 2009, pp. 11–18.
[14] Y. Cheng and X. Lin, "An edge-centric approach for change point detection in dynamic networks," in *Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining*. Cham, Switzerland: Springer, 2014, pp. 562–571.
[15] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *J. Amer. Soc. Inf. Sci.*, vol. 41, no. 6, pp. 391–407, 1990.
[16] H. Zha and H. D. Simon, "On updating problems in latent semantic indexing," *SIAM J. Sci. Comput.*, vol. 21, no. 2, pp. 782–791, 1999.
[17] J. E. Tougas and R. J. Spiteri, "Updating the partial singular value decomposition in latent semantic indexing," *Comput. Statist. Data Anal.*, vol. 52, no. 1, pp. 174–183, 2007.
[18] T. K. Landauer, D. S. McNamara, S. Dennis, and W. Kintsch, *Handbook of Latent Semantic Analysis*. London, U.K.: Psychology Press, 2013.
[19] G. H. Golub and C. F. Van Loan, *Matrix Computations*, vol. 3. Baltimore, MD, USA: JHU Press, 2012.
[20] Z. Zhang and H. Zha, "Structure and perturbation analysis of truncated SVDs for column-partitioned matrices," *SIAM J. Matrix Anal. Appl.*, vol. 22, no. 4, pp. 1245–1262, 2001.
[21] S. Axler, *Linear Algebra Done Right*. Cham, Switzerland: Springer, 1997.
[22] P. J. E. Vermeulen and D. P. Casasent, "Karhunen-Loeve techniques for optimal processing of time-sequential imagery," *Opt. Eng.*, vol. 30, no. 4, pp. 415–423, 1991.
[23] R. M. Larsen, "Lanczos bidiagonalization with partial reorthogonalization," in *Proc. DAIMI Rep. Ser.*, vol. 27, no. 537, pp. 1–101, 1998.
[24] K. Xu, "Stochastic block transition models for dynamic networks," *Artif. Intell. Statist.*, vol. 38, pp. 1079–1087, Feb. 2015.

[25] *Sample Document-By-Term Matricesand Term Lists*. Accessed: Sep. 20, 2018. [Online]. Available: http://web.eecs.utk.edu/research/lsi/corpa.html
[26] *PROPACK*. Accessed: Sep. 20, 2018. [Online]. Available: http://soi.stanford.edu/~rmunk/PROPACK/



YONGSHENG CHENG received the B.S. degree in communication engineering from the Huazhong University of Science and Technology, Wuhan, in 2009, and the Ph.D. degree in information and communication engineering from Tsinghua University, Beijing, in 2014. He is currently an Engineer with the Beijing Institute of Space Long March Vehicle. His research interests include wireless communication systems and data mining.



JIANG ZHU received the B.S. degree in electronic science and technology from Harbin Engineering University, Harbin, in 2011, and the Ph.D. degree in information and communication engineering from Tsinghua University, Beijing, in 2016. Since then, he has been a Lecturer with the Ocean College, Zhejiang University. His research interests include statistical signal processing and compressed sensing related topics.

XIAOKANG LIN received the B.E. degree from the Department of Electronics Engineering, Tsinghua University, Beijing, China, in 1970. Since then, he has been with Tsinghua University as an Assistant Teacher, a Lecturer, an Associate Professor, and currently a Full Professor. He was also a Visiting Scholar with Stuttgart University, Germany, in 1988, and The Hong Kong University, Hong Kong, in 1994. His research interests include high-speed switching, MPLS, broadband networks, and IC design.