On-Line Anomaly Detection With High Accuracy

Kun Xie, *Member, IEEE*, Xiaocan Li, Xin Wang, *Member, IEEE*, Jiannong Cao, *Fellow, IEEE*, Gaogang Xie, *Member, IEEE*, Jigang Wen, Dafang Zhang, *Member, IEEE*, and Zheng Qin

***Abstract*— Traffic anomaly detection is critical for advanced Internet management. Existing detection algorithms generally convert the high-dimensional data to a long vector, which compromises the detection accuracy due to the loss of spatial information of data. Moreover, they are generally designed based on the separation of normal and anomalous data in a time period, which not only introduces high storage and computation cost but also prevents timely detection of anomalies. Online and accurate traffic anomaly detection is critical but difficult to support. To address the challenge, this paper directly models the monitoring data in each time slot as a 2-D matrix, and detects anomalies in the new time slot based on bilateral principal component analysis (B-PCA). We propose several novel techniques in OnlineBPCA to support quick and accurate anomaly detection in real time, including a novel B-PCA-based anomaly detection principle that jointly considers the variation of both row and column principal directions for more accurate anomaly detection, an approximate algorithm to avoid using iteration procedure to calculate the principal directions in a close-form, and a sequential anomaly algorithm to quickly update principal directions with low computation and storage cost when receiving a new data matrix at a time slot. To the best of our knowledge, this is the first work that exploits 2-D PCA for anomaly detection. We have conducted extensive simulations to compare our OnlineBPCA with the state-of-art anomaly detection algorithms using real traffic traces Abilene and GÈANT. Our simulation results demonstrate that, compared with other algorithms, our OnlineBPCA can achieve significantly better detection performance with low false positive rate, high true positive rate, and low computation cost.**

***Index Terms*— Anomaly detection, on-line algorithm, bilateral PCA.**

I. Introduction

TRAFFIC anomalies, caused by sources such as flash crowds, denial-of-service attacks, port scans, and the spreading of worms, can have detrimental effects on network services. Detecting and diagnosing these anomalies are critical to both network operators and end users.

Existing efforts [1]–[14] on anomaly detection usually model the traffic monitoring data of a time slot as a vector and use a traffic matrix to record the traffic monitoring data of a period. In the example traffic matrix of Fig.1, each row denotes an OD (origin-destination) pair and each column denotes a time slot. As normal traffic data generally exhibit strong spatio-temporal correlations [2], [8], [9], the normal traffic matrix has low-rank. Moreover, as it is very costly for an attacker to compromise a large number of OD pairs for a long period of time, the anomalous data over time also form a sparse matrix. Based on the observations, to detect anomalies, existing studies usually separate the observed traffic data into two parts, a low-rank normal data matrix and a sparse outlier data matrix as shown in Fig.1. After the separation, the anomalies are detected and located from the outlier part.

The techniques applied for anomaly detection based on data separation include PCA [3], [5], [6], [8]–[12], Robust PCA [14], [15], bilinear factor matrix norm minimization [16], and recent Direct Robust Matrix Factorization (DRMF) [1], [17]). Detecting anomalies generally based on off-line learning, these methods require storing all the monitoring data within a period and operate on these data, which not only introduces high storage and computation cost but also prevents timely detection of anomalies.

It is essential to detect a sudden or unexpected change of the traffic behavior as soon as possible. Although very important, real-time anomaly detection is extremely difficult to achieve. It requires a light-weight algorithm to accurately and quickly identify whether the newly arriving data contain anomalies or not. Different from data separation, there are very limited studies on online anomaly detection. The work in [18] attempts to check the variation of PCA transformation between time slots to detect the anomaly. Although it is effective, designed based on conventional PCA that only operates over a vector of data, it still models the traffic data in each time slot

Manuscript received April 26, 2017; revised November 24, 2017 and March 5, 2018; accepted March 14, 2018; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor C. W. Tan. Date of publication April 26, 2018; date of current version June 14, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61572184, Grant 61725206, Grant 61472130, Grant 61472131, and Grant 61772191, in part by the Hunan Provincial Natural Science Foundation of China under Grant 2017JJ1010, in part by the Science and Technology Key Projects of Hunan Province under Grant 2015TP1004 and Grant 2016JC2012, in part by the U.S. ONR under Grant N00014-17-1-2730, in part by the NSF under Grant ECCS 1408247, Grant CNS 1526843, and Grant ECCS 1731238, and in part by the Open Project Funding of the CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, under Grant CASNDST201704. *(Corresponding author: Kun Xie.)*

K. Xie is with the College of Computer Science and Electronics Engineering, Hunan University, Changsha 410006, China, and also with the CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, and with the Department of Electrical and Computer Engineering, The State University of New York at Stony Brook, Stony Brook, NY 11794 USA (e-mail: xiekun@hnu.edu.cn).

X. Li, D. Zhang, and Z. Qin are with the College of Computer Science and Electronics Engineering, Hunan University, Changsha 410006, China (e-mail: hnulxc@hnu.edu.cn; dfzhang@hnu.edu.cn; zqin@hnu.edu.cn).

X. Wang is with the Department of Electrical and Computer Engineering, The State University of New York at Stony Brook, Stony Brook, NY 11794 USA (e-mail: x.wang@stonybrook.edu).

J. Cao is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong (e-mail: csjcao@comp.polyu.edu.hk).

G. Xie and J. Wen are with the Network Research Center, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: xie@ict.ac.cn; wenjigang@ict.ac.cn).

Digital Object Identifier 10.1109/TNET.2018.2819507

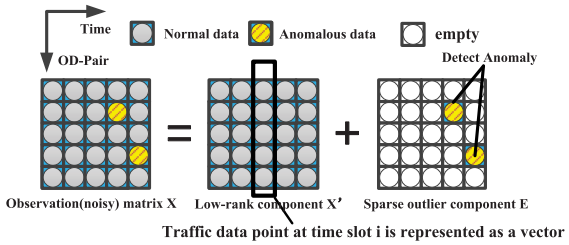


Fig. 1. Traffic anomaly detection methods based data separation.

as a vector. For a network consisting of N nodes, there are $N \times N$ OD pairs, which will form a very long vector of the size $N \times N$. This will in turn result in a large covariance matrix and large computation cost in the process of PCA transformation.

Actually, traffic monitoring data captured in one time slot can be naturally represented as an $N \times N$ matrix, with each row denoting the transmission data from the same origin node and each column denoting the transmission towards the same destination. In existing anomaly detection approaches, no matter through off-line data separation or online algorithm of [18], the data from network measurements in one time slot are usually modeled as a vector. The vector either directly contains all OD pairs as in Fig.1, or is formed with the concatenation of rows or columns of an $N \times N$ source-destination matrix through "matrix-to-vector alignment". Besides the computation complexity resulted from a long vector, the spatial information hidden in the traffic data will get lost when representing data in a vector form, which will compromise the performance of anomaly detection.

Rather than converting data from the matrix form to vector, directly using the matrix to represent traffic data in a time slot for anomaly detection can better capture the data relationship between rows and columns, which may help increase the detection accuracy. To understand the possibility and benefit, in this work, we propose to directly apply bilateral PCA (B-PCA) over the data points corresponding to two-dimensional data matrices for on-line anomaly detection. Although there are some recent studies [19]–[22] on the direct use of two-dimensional PCA over image data for feature extraction, these methods for feature extraction cannot be directly used for anomaly detection. Applying B-PCA for on-line anomaly detection faces three major challenges:

- There lacks a principle to exploit the data features from both rows and columns for anomaly detection;
- Different from conventional PCA, B-PCA does not have close-form solutions for finding the projection matrices, which makes it even harder to quickly detect the anomaly;
- Performing B-PCA operation over a huge set of historical data is computationally expensive and time consuming.

Different from conventional algorithms for anomaly detection, we model the network monitoring data in each time slot as a 2D matrix, and propose an *online* anomaly detection scheme based on B-PCA (termed OnlineBPCA) to detect whether newly arriving data contain anomalies or not. In addition, to enable online anomaly detection, we propose a set of algorithms for quick data processing. Our OnlineBPCA includes the following set of novel techniques to support quick and accurate anomaly detection:

- We propose a novel two-directional-change-based ***anomaly detection principle*** to detect whether newly arriving data contain anomalies, where we check the changes of the principal directions from both row side and column side. This is the first anomaly detection principle proposed that allows B-PCA to be applied for anomaly detection. As these two principal directions can more comprehensively and accurately extract the features hidden in the monitoring data, our OnlineBPCA can achieve a higher accuracy in detecting the anomaly than conventional anomaly detection algorithms.
- We propose an ***approximate algorithm*** to calculate the principal directions in a close form without using the iteration procedure, which in turn provides the possibility of designing a sequential algorithm for quickly detecting anomalies online. Our simulation results demonstrate that such an approximation does not decrease the detection accuracy while significantly reducing the computation cost.
- To quickly detect anomalous data, principal directions need to be updated to adapt to the network changes in real time. Unlike the batch methods which process all the data together, we propose a ***sequential anomaly detection algorithm*** that does not require the storage of the past data and can update the principal directions using the most recent monitoring data. As a result, our method is fast and preferred for streaming data and on-line anomaly detection.
- To amplify the impact of newly arriving data on the principal directions of the monitoring data set, we propose a novel ***strength method*** to duplicate the newly arriving data multiple times. This would make it easier to find anomalies even for a large data set.

Using traffic trace data Abilene [23] and GÈANT [24], we implement ten anomaly detection algorithms to evaluate our OnlineBPCA. Compared with other peer algorithms, OnlineBPCA can detect whether newly arriving data contain anomalies with much lower False Positive Rate and higher True Positive Rate. Specifically, benefiting from our approximate algorithm and sequential algorithm, OnlineBPCA can accurately detect the anomaly with very fast speed.

The rest of this paper is organized as follows. Section II presents the related work. We introduce the preliminary work on B-PCA in Section III. We describe our anomaly detection principle, approximate algorithm to find the projection matrices, and sequential anomaly detection algorithm in Section IV, Section V, and Section VI, respectively. Finally, we evaluate the performance using real traffic trace data in Section VIII, and conclude the work in Section IX.

II. Related Work

Principal Component Analysis (PCA) [7] is perhaps the best-known statistical analysis technique to achieve data separation for anomaly detection. PCA uses an orthogonal transformation to convert possibly correlated observed variables into a set of linearly uncorrelated variables (called principal

components or principal directions). Some recent papers that apply PCA to the traffic anomaly detection have shown some promising initial results [3], [5], [6], [8]–[12]. The principal components (PCs) are found and sorted in the order of contribution to overall variance, with the PCs in the lower dimensional and higher dimensional subspaces capturing the dynamic properties of the system and noisy information, respectively. As a result, to separate data, the PCs can be divided into two sets. The traffic data mapped to principal components (PCs) in the lower dimensional subspace are normal, and the remaining data are the anomalies. For data separation, all traffic data should be calculated and mapped to the two subspaces, corresponding respectively to the two types of PCs.

Although PCA-based data separation is effective when the corruption is caused by small additive noise, recent study [13] shows that traditional PCA-based approaches fail under the large corruption, even if the corruption affects only very few of the observations. To achieve better data separation, recently, Candès *et. al.* [14] propose Robust PCA (RPCA) which decomposes a given observation (noisy) matrix X into a low-rank component X' and a sparse outlier component E. To make the problem solvable, the work in [15] replaces the matrix rank and the cardinality ($\|\|\|_0$) functions with their convex surrogates, the nuclear norm $\|\|\|_*$ (i.e., the sum of its singular values) and the L_1 norm $\|\|\|_1$, and solves the following convex optimization problem

$$\min_{X',E} \ \{\|\mathrm{X}'\|_* + \lambda\|E\|_1\} \quad st.\ X' + E = X \tag{1}$$

where λ is a positive weighting parameter. To decompose the data into low-rank component and sparse component, these methods resort to some relaxation techniques which may largely impact the accuracy of anomaly detection.

To conquer the challenge in RPCA, work in [17] proposes Direct Robust Matrix Factorization (DRMF) which directly formulates the problem in its original way using the matrix rank to represent the low rank feature of normal data matrix and the L_0-norm to represent the sparse feature of the anomaly data. However, the solution involves the iterative execution of SVD decomposition, which will bring very high computation cost and is not scalable for large traffic data.

Shang *et al.* [16] observe two other issues of RPCA, which leads the solution to be biased. That is, the use of nuclear norm over penalizes large singular values, and the use of L_1 norm over penalizes large entries of the matrix. To address the issues, the authors propose two bilinear factor matrix norm minimization models for robust principal component analysis. Specifically, the paper considers two specific l_p-norm minimization, with $p = 1/2$ and $p = 2/3$, respectively.

Although data separation is an effective way of detecting anomalies appearing within a period, it is not suitable for online detection. Data separation approaches usually work off-line and require operating on the whole set of traffic data captured in a period consisting of multiple time slots, which consume a large amount of storage and long computation time. Moreover, some data separation techniques such as RPCA and DRMF need a time-consuming iterative process to separate the observed traffic data, which further increases the computation cost. In contrast, our anomaly detection algorithm aims to quickly and accurately detect whether traffic data newly arriving in a time slot contain anomalies.

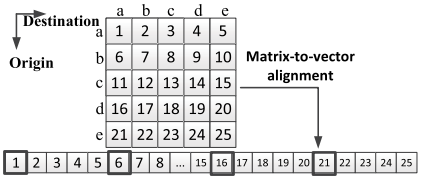


Fig. 2. Matrix-to-vector alignment.

Very limited work [18] studies the on-line anomaly detection. Different from PCA-based algorithms [5], [6], [8]–[11] [3], [12] that separate normal data and anomalous data by mapping the traffic data into two types of PCs, [18] proposes to detect whether the new data contain anomalies by checking the impact on PCs from the newly arriving data. However, as it is designed based on conventional PCA which can only operate vector-form based data points, the traffic monitoring data of each time slot in [18] is modeled as a vector like other traffic anomaly detection algorithms, which suffer from the loss of spatial data correlation and computation complexity.

As shown in Fig.2, the entries of first column denote the traffic data from different source nodes to the same destination node a, and are thus correlated. However, they are set far away from each other in the vectorized representation. A column anomaly can happen when a network of remotely controlled and widely scattered Zombies or Botnet computers launch DDoS attacks by simultaneously sending a large amount of traffic and/or a large number of service requests to the target system. Besides losing the spatial information and compromising the anomaly detection, "matrix-to-vector alignment" also leads the vector to be long, which will result in a large size covariance matrix in [18]'s approach. Computing the eigenvectors of a large covariance matrix is very time-consuming.

Different from conventional PCA which can only deal with vector data, a new technique called two-dimensional principal component analysis (2DPCA) [19] was recently proposed in the image field to handle 2D image, which directly computes eigenvectors of the so-called covariance matrix without matrix-to-vector conversion. Because the size of the covariance matrix is equal to the width of images, which is quite small compared with the size of a covariance matrix in PCA, 2DPCA evaluates the image covariance matrix more accurately and computes the corresponding eigenvectors more efficiently than PCA. Based on tests over data from several databases [19], the accuracy of face recognition using 2DPCA was found to be higher than that using PCA, and the extraction of image features is computationally more efficient.

However, studies in [26] show that 2DPCA is essentially working in the row direction of images. If we apply 2DPCA to anomaly detection, the information hidden in the data matrix can not be fully utilized, so the detection accuracy will be still low. To overcome the restrictions under 2DPCA, two linear

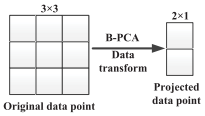


Fig. 3. Illustration of B-PCA.

transforms can be applied to both the left and the right side of the input image matrices. Recent studies [20], [22] propose to apply B-PCA to perform projections in both row and column modes for better dimensionality reduction.

Different from current traffic anomaly detection approaches, we directly model the monitoring data in each time slot as a 2D matrix and propose an anomaly detection algorithm which allows the use of bilateral PCA (B-PCA) to preserve the spatial correlations among data for more accurate anomaly detection. Specially, we are the first to design the detection principle that B-PCA can be exploited for anomaly detection. Moreover, to support quick anomaly detection, we propose an approximate algorithm to calculate the projection directions in a close form and also an sequential anomaly detection algorithm that does not require the storage of the past data and can update the principal directions using the most recent monitoring data.

III. Preliminary Work on B-PCA

In this section, we firstly introduce the basic concept of B-PCA, then the iteration algorithm to calculate the projection matrices in B-PCA, and finally analyze the challenges in applying B-PCA to on-line anomaly detection.

A. B-PCA

B-PCA transforms a high-dimensional $M \times N$ data to a low dimensional $r \times l$ projected data with $r < M, l < N$ to abstract the hidden features of the data matrix. Fig.3 shows an example of B-PCA. In this example, B-PCA transforms a 3×3 matrix to a 2×1 projected data matrix. To achieve such a transformation, B-PCA applies two projection matrices, i.e., a left (column) projection matrix $\mathbf{U} = \{u_1, \cdots, u_r\} \in R^{M\times r}$ and a right (row) projection matrix $\mathbf{V} = \{v_1, \cdots, v_l\} \in R^{N\times l}$. $u_i\,(1 \le i \le r)$ and $v_j\,(1 \le j \le l)$ are principal directions (components), which are orthogonal to each other. That is, $u_i^T u_j = 0$ and $v_i^T v_j = 0$ if $i \ne j$. As we will shown in Section IV, for anomaly detection, only the direction of the principal component is important, so we have $\|u_i\|_2 = 1$, and $\|v_i\|_2 = 1$.

Given the data set $\mathcal{X}_t = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_t\}$ with $\mathbf{X}_i \in R^{M\times N}, 1 \le i \le t$, we have the data set

$$\begin{aligned}\mathcal{Y}_t &= \left\{\mathbf{X}_1 - \bar{\mathbf{X}}_t, \mathbf{X}_2 - \bar{\mathbf{X}}_t, \cdots, \mathbf{X}_t - \bar{\mathbf{X}}_t\right\} \\ &= \{\mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_t\} \end{aligned} \tag{2}$$

after data centering, where $\mathbf{Y}_i = \mathbf{X}_i - \bar{\mathbf{X}}_t$ and the global mean of this data set $\bar{\mathbf{X}}_t = \frac{1}{t}\sum_{i=1}^{t}\mathbf{X}_i$. The projected data point of $\mathbf{Y}_i$ is $\mathbf{Z}_i = \mathbf{U}^T\mathbf{Y}_i\mathbf{V}$. On the other hand, with $\mathbf{Z}_i$, the data point in the original high dimensional space can be found as $\mathbf{U}\mathbf{Z_i}\mathbf{V^T}$. As $r < M$ and $l < N$, the data information may be lost during the projections. To obtain the optimal projection matrices, B-PCA tries to minimize the information loss from the projections:

$$\min_{\mathbf{U},\mathbf{V}} \sum_{i=1}^{t} \left\|\mathbf{Y}_i - \mathbf{U}\mathbf{Z}_i\mathbf{V}^T\right\|_F^2 \tag{3}$$

where t is the number of data points and $\|\bullet\|_F$ is the Frobenius norm of a matrix.

B. Iteration Algorithm to Calculate the Projection Matrices

Obtaining the projection matrices $\mathbf{U}$ and $\mathbf{V}$ by directly solving the problem in Eq. (3) is difficult. According to [20, Th. 2], problem defined in Eq.(3) is equivalent to the following problem

$$\max_{\mathbf{U},\mathbf{V}} \sum_{i=1}^{t} \left\|\mathbf{U}^T\mathbf{Y}_i\mathbf{V}\right\|_F^2 \tag{4}$$

However, there is no close-form solution for problem in (4). Reference [20] provides iterative solution to solve the problem. Given a data set $\mathcal{Y}_t = \{\mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_t\}$ and the column projection matrix $\mathbf{U}_{opt}$, the problem (4) can be further written as $\max_{\mathbf{V}} \sum_{i=1}^{t} \left\|\mathbf{U}_{opt}^T\mathbf{Y}_i\mathbf{V}\right\|_F^2$. As

$$\begin{aligned}\left\|\mathbf{U}_{opt}^T\mathbf{Y}_i\mathbf{V}\right\|_F^2 &= tr\left(\left(\mathbf{U}_{opt}^T\mathbf{Y}_i\mathbf{V}\right)^T\left(\mathbf{U}_{opt}^T\mathbf{Y}_i\mathbf{V}\right)\right) \\ &= tr\left(\mathbf{V}^T\left(\mathbf{U}_{opt}^T\mathbf{Y}_i\right)^T\left(\mathbf{U}_{opt}^T\mathbf{Y}_i\right)\mathbf{V}\right), \end{aligned} \tag{5}$$

if we denote the covariance matrix

$$\mathbf{C}_t^v = \sum_{i=1}^{t}\left(\mathbf{U}_{opt}^T\mathbf{Y}_i\right)^T\left(\mathbf{U}_{opt}^T\mathbf{Y}_i\right), \tag{6}$$

the problem in (4) can be further written as $\max_{\mathbf{V}} tr\left(\mathbf{V}^T\mathbf{C}_t^{\mathrm{v}}\mathbf{V}\right)$. To solve this problem, the row projection matrix $\mathbf{V}$ can be formed by the first l eigenvectors corresponding to the first l largest eigenvalues of $\mathbf{C}_t^v$. Similarly, given the row projection matrix $\mathbf{V}_{opt}$, the column projection matrix U can be formed by the first r eigenvectors corresponding to the first r largest eigenvalues of the covariance matrix $\mathbf{C}_t^u = \sum_{i=1}^{t}(\mathbf{Y}_i\mathbf{V}_{opt})(\mathbf{Y}_i\mathbf{V}_{opt})^T$.

According to the above process, Algorithm 1 [20] is proposed to calculate the projection matrices $\mathbf{U}$ and $\mathbf{V}$ iteratively until the changes of $\mathbf{U}$ and $\mathbf{V}$ between two consecutive steps are very small.

C. Challenges in Applying B-PCA to Anomaly Detection

It is more efficient to perform B-PCA over the matrix than conventional PCA over a long vector. However, there are two major challenges to apply B-PCA for on-line anomaly detection:

- ***How to apply B-PCA for anomaly detection?*** All existing 2-dimensional PCA approaches are proposed and implemented to extract image features. It is unclear how the data should be processed and what principles to follow for B-PCA to determine the existence of anomalies.
- ***How to enable on-line anomaly detection?*** Unlike conventional PCA, there is no close-form equation to

Algorithm 1 Finding the Projection Matrices Iteratively

Input: $\mathcal{X} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_t\}$

Output: column projection matrix $\mathbf{U}$, row projection matrix $\mathbf{V}$

1: Data centering: $\mathcal{Y}_t = \{\mathbf{X}_1 - \bar{\mathbf{X}}_t, \mathbf{X}_2 - \bar{\mathbf{X}}_t, \cdots, \mathbf{X}_t - \bar{\mathbf{X}}_t\} = \{\mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_t\}$ where $\bar{\mathbf{X}}_t$ is the global mean of data set $\mathcal{X}$.

2: Initialize $k = 0$, $(\mathbf{V})_k = \mathbf{I}$

3: **while** not converged **do**

4: $k = k + 1$

5:

$$(\mathbf{C}^u)_k = \sum_{i=1}^{t} (\mathbf{Y}_i (\mathbf{V})_{k-1}) (\mathbf{Y}_i (\mathbf{V})_{k-1})^T \tag{7}$$

6: Set $(\mathbf{U})_k$ as the matrix formed by the first r eigenvectors corresponding to the first r largest eigenvalues of the covariance matrix $(\mathbf{C}^u)_k$.

7:

$$(\mathbf{C}^v)_k = \sum_{i=1}^{t} \left((\mathbf{U}^T)_k \mathbf{Y}_i \right)^T \left((\mathbf{U}^T)_k \mathbf{Y}_i \right) \tag{8}$$

8: Set $(\mathbf{V})_k$ as the matrix formed by the first l eigenvectors corresponding to the first l largest eigenvalues of the covariance matrix $(\mathbf{C}^v)_k$.

9: **end while**

10: $\mathbf{U} = (\mathbf{U})_{\mathbf{k}}$, $\mathbf{V} = (\mathbf{V})_{\mathbf{k}}$, return $\mathbf{U}$ and $\mathbf{V}$

calculate the projection matrices in current B-PCA studies. Using an iterative procedure in Algorithm 1 for matrix projection would incur a large computation cost. Moreover, iterative methods are difficult to apply for processing data sequentially. These prevent B-PCA from being directly used for online anomaly detection.

IV. Anomaly Detection Principle Based on B-PCA

For a network consisting of N nodes, we define a monitoring data matrix, $\mathbf{X}_t \in R^{N \times N}$, to hold the end-to-end monitoring data at time slot t, with the (ij)-th entry, $\mathbf{X}_t(ij)$, representing the monitoring data from node i to node j captured in time slot t. As shown in Fig. 4, the monitoring data set $\{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_k, \cdots, \mathbf{X}_t\}$ forms a tensor with each slice $\mathbf{X}_k \in R^{N \times N}, 1 \leq k \leq t$ being the monitoring data of one time slot. Given the set of monitoring data already collected up to the time slot t, $\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_t$, our goal is to quickly and accurately detect whether the newly coming data $\mathbf{X}_{t+1}$ is anomalous or not.

We apply B-PCA to transform the high-dimensional $N \times N$ monitoring data to low dimensional $r \times l$ projected data with $r < N, l < N$. Specially, according to B-PCA data transformation in Eq.(3), the low dimensional principal space (defined by the projection matrices $\mathbf{U}$ and $\mathbf{V}$) should be found to minimize the information loss between the data points and their projections. If there are anomalies added to a data set, the principal subspace found by B-PCA will be remarkably affected, with the principal directions significantly changed.

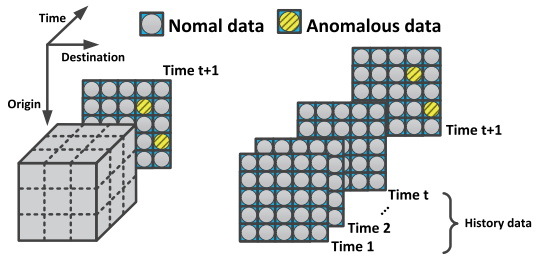


Fig. 4. Illustration of the online anomaly detection problem.

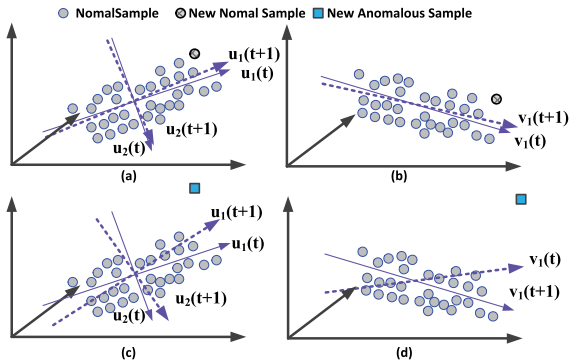


Fig. 5. Anomaly detection using B-PCA. (a) Left Subspace (add normal sample). (b) Right Subspace (add normal sample). (c) Left Subspace (add anomalous sample). (d) Right Subspace (add anomalous sample).

Following the example in Fig.3, B-PCA transforms 3×3 data points to 2×1 projected data points. As shown in Fig.5, the left projection matrix $\mathbf{U}$ has 2 principal components u_1, u_2 while the right projection matrix $\mathbf{V}$ has only 1 principal component v_1. The clustered gray circles represent normal data points, and the blue square denotes an outlier. In this example, when an outlier point is added, the principal directions (both the row principal directions Fig.5(c) and the column principal directions Fig.5(d)) should deviate toward the outlier. This is because the subspaces are found through the minimization of the information loss between all the data points and their corresponding projections. More specifically, the presence of such an outlier data point produces a large angle between the resulting and the original principal directions. On the other hand, this angle will be small when a normal data point is added (Fig.5(a) and Fig.5(b)).

Therefore, we propose an **anomaly detection principle**: To detect if the data point collected at time $t+1$ is anomalous, there is a need to check if the row and column principals have big direction changes. To exploit both directional changes, we propose to use the joint projection matrices from consecutive time slots t and $t+1$: $\mathbf{M}_t = [\mathbf{U}_t, \mathbf{V}_t]$, $\mathbf{M}_{t+1} = [\mathbf{U}_{t+1}, \mathbf{V}_{t+1}]$, where $\mathbf{U}_t$ and $\mathbf{V}_t$ are the projection matrices before getting the data $\mathbf{X}_{t+1}$, $\mathbf{U}_{t+1}$ and $\mathbf{V}_{t+1}$ are the projection matrices after obtaining $\mathbf{X}_{t+1}$. We define a metric

$$Cosine = \left| \frac{\langle Vec(\mathbf{M}_t), Vec(\mathbf{M}_{t+1}) \rangle}{\|Vec(\mathbf{M}_t)\| \, \|Vec(\mathbf{M}_{t+1})\|} \right| \tag{9}$$

to capture the influence of the newly added monitoring data on the principal directions. In (9), $Vec(\mathbf{A})$ is the vectorization operation of a matrix $\mathbf{A}$. A smaller Cosine corresponds to larger direction changes.

Accordingly, if Cosine is less than a threshold, we will identify that the newly added monitoring data $\mathbf{X}_{t+1}$ is an anomaly. In the simulation, we will use the trace data to train the threshold.

Algorithm 2 Anomaly Detection Based on B-PCA

Input: The newly arriving monitoring data $\mathbf{X}_{t+1}$
The history data set $\mathcal{X} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_t\}$
The projection matrices of history data $\mathbf{U}_t$,$\mathbf{V}_t$
Output: The projection matrices of current data $\mathbf{U}_{t+1}$,$\mathbf{V}_{t+1}$;
Identify whether $\mathbf{X}_{t+1}$ is anomalous data or not
1: Add $\mathbf{X}_{t+1}$ to the data set and obtain the update data set $\mathcal{X} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_t, \mathbf{X}_{t+1}\}$, calculate the average of the update data set, denoted as $\bar{\mathbf{X}}_{t+1}$
2: Data centering:
$\mathcal{Y}_{t+1} = \{\mathbf{X}_1 - \bar{\mathbf{X}}_{t+1}, \mathbf{X}_2 - \bar{\mathbf{X}}_{t+1}, \cdots, \mathbf{X}_t - \bar{\mathbf{X}}_{t+1}, \mathbf{X}_{t+1} - \bar{\mathbf{X}}_{t+1}\} = \{\mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_t, \mathbf{Y}_{t+1}\}$.
3: Apply the iterative algorithm in Algorithm 1 to the centering data set $\mathcal{Y}_{t+1}$ to calculate $\mathbf{V}_{t+1}$ and $\mathbf{U}_{t+1}$.
4: Build the combination projection matrices $\mathbf{M}_t = [\mathbf{U}_t, \mathbf{V}_t]$, $\mathbf{M}_{t+1} = [\mathbf{U}_{t+1}, \mathbf{V}_{t+1}]$

$$Cosine = \left| \frac{\langle Vec(\mathbf{M}_t), Vec(\mathbf{M}_{t+1}) \rangle}{\|Vec(\mathbf{M}_t)\| \, \|Vec(\mathbf{M}_{t+1})\|} \right| \tag{10}$$

5: **if** $Cosine \geq score$ **then**
6: $\mathbf{X}_{t+1}$ is not anomaly data
7: **else**
8: $\mathbf{X}_{t+1}$ is anomaly data
9: $\mathbf{U}_{t+1} = \mathbf{U}_t$,$\mathbf{V}_{t+1} = \mathbf{V}_t$
10: **end if**

V. Approximate Algorithm to Find the Two Projection Matrices

Although anomaly detection can be performed based on B-PCA following Algorithm 2, it involves a computation intensive and time-consuming iterative process in Algorithm 1 to find the projection matrices using all the traffic data. Thus Algorithm 2 is not applicable for on-line processing, or operating over a big data set. In this work, we propose an approximate algorithm to quickly calculate the projection matrices.

In Algorithm 1, the two projection matrices $\mathbf{U}_k$ and $\mathbf{V}_k$ (on lines 6, 8 of the algorithm) are determined based on the covariance matrices

$$(\mathbf{C}^u)_k = \sum_{i=1}^{t} (\mathbf{Y}_i (\mathbf{V})_{k-1}) (\mathbf{Y}_i (\mathbf{V})_{k-1})^T \tag{11}$$

$$(\mathbf{C}^v)_k = \sum_{i=1}^{t} \left((\mathbf{U}^T)_k \mathbf{Y}_i \right)^T \left((\mathbf{U}^T)_k \mathbf{Y}_i \right) \tag{12}$$

which further depend on the $(\mathbf{V})_{k-1}$ and $(\mathbf{U})_k$ obtained from the previous $(k-1)_{th}$ and current $(k)_{th}$ iteration steps. Thus Algorithm 1 has to be run iteratively.

To investigate the convergence behavior of Algorithm 1, we run Algorithm 1 over the public traffic traces Abilene [23] and GÈANT [24]. In Fig.6, y axis shows the gaps $\mathbf{U}_{gap}$ and $\mathbf{V}_{gap}$ of projection matrices calculated between two consecutive steps, where $\mathbf{U}_{gap} = \frac{\|\mathbf{U}_k - \mathbf{U}_{k-1}\|_F}{\|\mathbf{U}_k\|_F}$ and $\mathbf{V}_{gap} = \frac{\|\mathbf{V}_k - \mathbf{V}_{k-1}\|_F}{\|\mathbf{V}_k\|_F}$. We can see Algorithm 1 converges at the iteration step 10 for both two traffic traces. As each iteration in Algorithm 1 requires performing eigen decomposition twice (corresponding to steps 6 and 8), the overall computation cost is even higher.

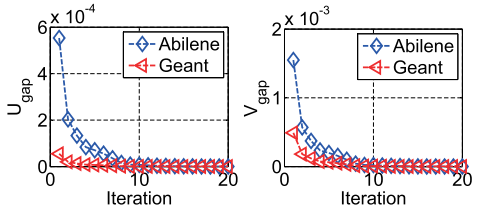


Fig. 6. Convergence behavior of Algorithm 1.

Following, we first derive the approximate covariance matrices, and then propose our approximate algorithm to calculate projection matrices directly in a close form without involving iteration.

Approximate Covariance Matrix $\mathbf{C}_t^u$*:* In (11), as $(\mathbf{V})_{k-1}$ is the matrix formed by the first l eigenvectors corresponding to the first l largest eigenvalues of the covariance matrix $(\mathbf{C}^v)_{k-1}$, if $l = N$, we have $(\mathbf{V})_{k-1}(\mathbf{V}^T)_{k-1} = I$. In this case, $(\mathbf{C}^u)_k = \sum_{i=1}^{t} (\mathbf{Y}_i(\mathbf{V})_{k-1}) (\mathbf{Y}_i(\mathbf{V})_{k-1})^T = \sum_{i=1}^{t} \mathbf{Y}_i \left((\mathbf{V})_{k-1}(\mathbf{V}^T)_{k-1} \right) \mathbf{Y}_i^T = \sum_{i=1}^{t} \mathbf{Y}_i \mathbf{Y}_i^T$. As l is set to keep as much original information as possible, the data loss during the data transformation from $\mathbf{Y}_i$ to $\mathbf{Y}_i(\mathbf{V})_{k-1}(\mathbf{V}^T)_{k-1}$ is very small. Thus we have $\mathbf{Y}_i \approx \mathbf{Y}_i(\mathbf{V})_{k-1}(\mathbf{V}^T)_{k-1}$, based on which, we can further deduce that

$$(\mathbf{C}^u)_k = \sum_{i=1}^{t} (\mathbf{Y}_i(\mathbf{V})_{k-1}) (\mathbf{Y}_i(\mathbf{V})_{k-1})^T \approx \sum_{i=1}^{t} \mathbf{Y}_i \mathbf{Y}_i^T \tag{13}$$

Approximate Covariance Matrix $\mathbf{C}_t^v$*:* Similarly, in (12), $(\mathbf{U})_k$ is the matrix formed by the first r eigenvectors corresponding to the first r largest eigenvalues of the covariance matrix $(\mathbf{C}^u)_k$. As r is set to keep as much original information as possible, the data loss during the data transformation from $\mathbf{Y}_i^T$ to $\mathbf{Y}_i^T(\mathbf{U})_k(\mathbf{U}^T)_k$ is very small. Thus we have $\mathbf{Y}_i^T \approx \mathbf{Y}_i^T(\mathbf{U})_k(\mathbf{U}^T)_k$, based on which, we can deduce that

$$(\mathbf{C}^v)_k = \sum_{i=1}^{t} \left((\mathbf{U}^T)_k \mathbf{Y}_i \right)^T \left((\mathbf{U}^T)_k \mathbf{Y}_i \right) \approx \sum_{i=1}^{t} \mathbf{Y}_i^T \mathbf{Y}_i \tag{14}$$

We denote $\mathbf{C}_t^u = \sum_{i=1}^{t} \mathbf{Y}_i \mathbf{Y}_i^T$ and $\mathbf{C}_t^v = \sum_{i=1}^{t} \mathbf{Y}_i^T \mathbf{Y}_i$. Fortunately, covariance matrices $\mathbf{C}_t^u$ and $\mathbf{C}_t^v$ only depend on the centralized data $\mathbf{Y}_i$ and $\mathbf{Y}_i^T$, which provides the possibility to derive close form projection matrices.

Approximate Algorithm: Although we can derive projection matrices $\mathbf{U}_t$ and $\mathbf{V}_t$ directly using $\mathbf{C}_t^u$ and $\mathbf{C}_t^v$ through eigen decomposition, to facilitate designing a sequential algorithm in Section VI-C, we propose to derive $\mathbf{U}_t$ and $\mathbf{V}_t$ without calculating the summation of $\sum_{i=1}^{t} \mathbf{Y}_i \mathbf{Y}_i^T$ and $\sum_{i=1}^{t} \mathbf{Y}_i^T \mathbf{Y}_i$ in $\mathbf{C}_t^u$ and $\mathbf{C}_t^v$.

Let $\mathbf{Y}_t^v = ((\mathbf{X}_1 - \bar{\mathbf{X}}_t)^T, \cdots, (\mathbf{X}_t - \bar{\mathbf{X}}_t)^T) \in R^{N \times tN}$, and $\mathbf{Y}_t^u = ((\mathbf{X}_1 - \bar{\mathbf{X}}_t), \cdots, (\mathbf{X}_t - \bar{\mathbf{X}}_t)) \in R^{N \times tN}$. Obviously, $\mathbf{Y}_t^v$ and $\mathbf{Y}_t^u$ are large matrices formed with the concatenation of

data points from different time slots. We have

$$\mathbf{C}_t^v = \sum_{i=1}^{t} \mathbf{Y}_i^T \mathbf{Y}_i = \sum_{i=1}^{t} \left(\mathbf{X}_i - \bar{\mathbf{X}}_t\right)^T \left(\mathbf{X}_i - \bar{\mathbf{X}}_t\right) = \mathbf{Y}_t^v (\mathbf{Y}_t^v)^T,$$
$$\mathbf{C}_t^u = \sum_{i=1}^{t} \mathbf{Y}_i \mathbf{Y}_i^T = \sum_{i=1}^{t} \left(\mathbf{X}_i - \bar{\mathbf{X}}_t\right) \left(\mathbf{X}_i - \bar{\mathbf{X}}_t\right)^T = \mathbf{Y}_t^u (\mathbf{Y}_t^u)^T. \tag{15}$$

Denoting the singular value decomposition of $\mathbf{Y}_t^u$ as $\mathbf{U}_t^u \mathbf{\Sigma}_t^u (\mathbf{V}_t^u)^T$, for the covariance matrix $\mathbf{C}_t^u = \mathbf{Y}_t^u (\mathbf{Y}_t^u)^T$, we have

$$\begin{aligned} \mathbf{C}_t^u &= \mathbf{Y}_t^u (\mathbf{Y}_t^u)^T = \left(\mathbf{U}_t^u \mathbf{\Sigma}_t^u (\mathbf{V}_t^u)^T\right) \left(\mathbf{U}_t^u \mathbf{\Sigma}_t^u (\mathbf{V}_t^u)^T\right)^T \\ &= \mathbf{U}_t^u \mathbf{\Sigma}_t^u (\mathbf{V}_t^u)^T \mathbf{V}_t^u \mathbf{\Sigma}_t^u \mathbf{U}_t^u = \mathbf{U}_t^u (\mathbf{\Sigma}_t^u)^2 (\mathbf{U}_t^u)^T. \end{aligned} \tag{16}$$

The last item $\mathbf{U}_t^u (\mathbf{\Sigma}_t^u)^2 (\mathbf{U}_t^u)^T$ is written in the format of eigenvalue decomposition of $\mathbf{C}_t^u$, that is $\mathbf{U}_t^u (\mathbf{\Sigma}_t^u)^2 (\mathbf{U}_t^u)^T = eig\left(\mathbf{C}_t^u\right)$. Thus, the eigenvalue decomposition of $\mathbf{C}_t^u$ is equivalent to the multiplication of the SVD of $\mathbf{Y}_t^u$. The eigen-vector of $\mathbf{C}_t^u$ is the left singular vector of $\mathbf{Y}_t^u$, with the eigen-values corresponding to the first r singular values of $\mathbf{Y}_t^u$. Similarly, the eigen-vector of $\mathbf{C}_t^v$ is the left singular vector of $\mathbf{Y}_t^v$, with the eigen-values corresponding to the first l singular values of $\mathbf{Y}_t^v$.

Therefore, the column projection matrix $\mathbf{U}_t$ can be calculated as the matrix formed by the first r left singular vectors corresponding to the first r singular values of the $\mathbf{Y}_t^u$; the row projection matrix $\mathbf{V}_t$ can be approximately calculated as the matrix formed by the first l left singular vectors corresponding to the first l largest eigenvalues of $\mathbf{Y}_t^v$.

VI. Sequential Anomaly Detection

Although our approximation algorithm can calculate the projection matrices without involving iterations, it operates on $\mathbf{Y}_t^v$ and $\mathbf{Y}_t^u$ over a large amount of historical data, which is still time consuming and not scalable. To timely detect the anomaly, we propose a sequential detection algorithm based only on the statistic values from the history data and the data samples taken in the new time slot. In this section, before we present the algorithm, we first investigate the relationship between covariance matrices obtained in two sequential monitoring steps.

A. Relationship Between Covariance Matrices Obtained in Two Sequential Monitoring Steps

When new monitoring data $\mathbf{X}_{\text{new}} = \mathbf{X}_{\text{t+1}}$ is coming, the monitoring data set is updated to be $\mathcal{X}_{\text{t+1}} = \{\mathbf{X}_1, \cdots, \mathbf{X}_{\text{t}}, \mathbf{X}_{\text{t+1}}\}$.

Theorem 1: The global mean of the update data set $\mathcal{X}_{\text{t+1}}$ is $\bar{\mathbf{X}}_{t+1} = \frac{1}{t+1}\left(t\bar{\mathbf{X}}_t + \mathbf{X}_{t+1}\right)$, the covariance matrix of the update data set are

$$\mathbf{C}_{\text{t+1}}^{\text{u}} = C_{\text{t}}^{\text{u}} + \frac{t}{(t+1)} \left(\bar{\mathbf{X}}_t - \mathbf{X}_{t+1}\right) \left(\bar{\mathbf{X}}_t - \mathbf{X}_{t+1}\right)^T \tag{17}$$

and

$$\mathbf{C}_{\text{t+1}}^{\text{v}} = C_{\text{t}}^{\text{v}} + \frac{t}{(t+1)} \left(\bar{\mathbf{X}}_t - \mathbf{X}_{t+1}\right)^T \left(\bar{\mathbf{X}}_t - \mathbf{X}_{t+1}\right) \tag{18}$$

Proof: Due to the limited space, the proof is omitted. ■

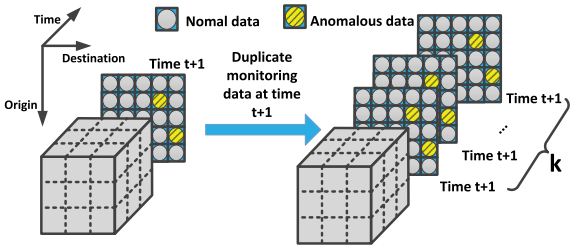


Fig. 7. Illustration of Strengthen method.

From Theorem 1, the covariance matrix (i.e.,$\mathbf{C}_{t+1}^u$ or $\mathbf{C}_{t+1}^v$) of the current updated data set can be calculated directly using the covariance matrix (i.e.,$\mathbf{C}_t^u$ or $\mathbf{C}_t^v$) of history data and the newly arriving data $\mathbf{X}_{t+1}$ through (17) and (18). Thus we do not need to store the history data and perform computation over the whole data set, and the space and computation cost can be largely reduced.

B. Strengthening Method

For practical anomaly detection, the size of the data set is typically large, and thus it might not be easy to observe the variation of principal directions caused by the presence of a single outlier. In this section, we introduce a strengthening method, and discuss how and why we are able to detect the existence of abnormal data instances according to the changes of principal directions, even in the presence of a large amount of data.

To amplify the impact of newly collected data point on the principal directions of the monitoring data set, we can duplicate the new data point multiple times in the data set, as shown in Fig.7.

Next, we will show that the strengthening method can effectively amplify the effect of outlier data compared to that of the normal data. Given the monitoring data set $\mathcal{X}_t = \{\mathbf{X}_1, \cdots, \mathbf{X}_i, \cdots, \mathbf{X}_{\text{t}}\}$ with $\mathbf{X}_i \in \mathrm{R}^{N \times N}$, when a new data point (i.e., $\mathbf{X}_{\text{new}} = \mathbf{X}_{\text{t+1}}$) arrives, after repeatedly adding this data point k times into the dataset, the dataset becomes

$$\mathcal{X}_{\text{t+1}} = \left\{ \underbrace{\mathbf{X}_1, \cdots, \mathbf{X}_{\text{t}}}_{t}, \underbrace{\mathbf{X}_{t+1}, \cdots, \mathbf{X}_{\text{t+1}}}_{k} \right\}. \tag{19}$$

According to the global mean and the covariance matrix definition, we have

$$\begin{aligned} \bar{\mathbf{X}}_{t+1} &= \frac{1}{t+k} \sum_{i=1}^{t+k} \mathbf{X}_i = \frac{1}{t+k} \left(\sum_{i=1}^{t} \mathbf{X}_i + \sum_{i=t+1}^{t+k} \mathbf{X}_i \right) \\ &= \frac{1}{t+k} \left(t\bar{\mathbf{X}}_t + k\mathbf{X}_{t+1}\right) \end{aligned} \tag{20}$$

and

$$\mathbf{C}_{t+1}^u = \mathbf{C}_t^u + \frac{tk}{(t+k)} \left(\bar{\mathbf{X}}_{\text{t}} - \mathbf{X}_{t+1}\right) \left(\bar{\mathbf{X}}_{\text{t}} - \mathbf{X}_{t+1}\right)^T \tag{21}$$

and

$$\mathbf{C}_{t+1}^v = \mathbf{C}_t^v + \frac{tk}{(t+k)} \left(\bar{\mathbf{X}}_{\text{t}} - \mathbf{X}_{t+1}\right)^T \left(\bar{\mathbf{X}}_{\text{t}} - \mathbf{X}_{t+1}\right) \tag{22}$$

There is a tradeoff to set the parameter k. A large k can increase the accuracy in detecting the outlier data by enlarging the gaps between $\mathbf{C}_{t+1}^u$ and $\mathbf{C}_t^u$, $\mathbf{C}_{t+1}^v$ and $\mathbf{C}_t^v$, respectively, but it also increases the chance of falsely detecting a normal data point as the outlier. On the other hand, a small k will make it difficult to detect the newly added outlier if the set of history data is large. Instead of setting k as a constant, we set k as $k = \alpha \cdot t$, where t is the number of history data points, α is the over-sampling ratio. We will investigate how the parameter α impacts the detection performance through simulations in Section VIII-C.3, and set the parameter α to an empirical value.

To increase the effectiveness of data amplification, we only duplicate the new data point when it just arrives in the data set, but will only keep one copy of this data point when checking the next arrival one.

C. A Complete Algorithm With Further Speedup

It is clear that, when a new monitoring data matrix is duplicately added into the monitoring data set, we have $\mathbf{C}_{t+1}^u = \mathbf{C}_t^u + \frac{tk}{(t+k)}\left(\bar{\mathbf{X}}_t - \mathbf{X}_{t+1}\right)\left(\bar{\mathbf{X}}_t - \mathbf{X}_{t+1}\right)^T$ in (21) and $\mathbf{C}_{t+1}^v = \mathbf{C}_t^v + \frac{tk}{(t+k)}\left(\bar{\mathbf{X}}_t - \mathbf{X}_{t+1}\right)^T\left(\bar{\mathbf{X}}_t - \mathbf{X}_{t+1}\right)$ in (22).

In Section V, we observe $\mathbf{C}_t^v = \mathbf{Y}_t^v(\mathbf{Y}_t^v)^T$ and $\mathbf{C}_t^u = \mathbf{Y}_t^u(\mathbf{Y}_t^u)^T$, which provides a solution to derive the projection matrices $\mathbf{U}_t$ and $\mathbf{V}_t$ through the decomposition of matrices $\mathbf{Y}_t^v$ and $\mathbf{Y}_t^u$ instead of covariance matrices $\mathbf{C}_t^u$ and $\mathbf{C}_t^v$. To further reduce the computation cost, we would like to to reuse the decomposition results of $\mathbf{Y}_t^v$ and $\mathbf{Y}_t^u$ to deduce the decomposition results of $\mathbf{Y}_{t+1}^v$ and $\mathbf{Y}_{t+1}^u$ so that we can make quick sequential processing for fast anomaly detection.

As $\mathbf{C}_t^v = \mathbf{Y}_t^v(\mathbf{Y}_t^v)^T$ and $\mathbf{C}_t^u = \mathbf{Y}_t^u(\mathbf{Y}_t^u)^T$, to make $\mathbf{C}_{t+1}^u = \mathbf{Y}_{t+1}^u\left(\mathbf{Y}_{t+1}^u\right)^T$ and $\mathbf{C}_{t+1}^v = \left(\mathbf{Y}_{t+1}^v\right)^T\mathbf{Y}_{t+1}^v$, we have

$$\mathbf{Y}_{t+1}^u = \left[\mathbf{Y}_t^u, \sqrt{tk/\left(t+k\right)}\left(\overline{\mathbf{X}}_t - \mathbf{X}_{t+1}\right)\right] \tag{23}$$

and

$$\mathbf{Y}_{t+1}^v = \left[\mathbf{Y}_t^v, \sqrt{tk/\left(t+k\right)}\left(\overline{\mathbf{X}}_t^T - \mathbf{X}_{t+1}^T\right)\right]. \tag{24}$$

where $\mathbf{Y}_{t+1}^u$ is a matrix by matrices $\mathbf{Y}_t^u$ and $\sqrt{tk/\left(t+k\right)}\left(\overline{\mathbf{X}}_t - \mathbf{X}_{t+1}\right)$, $\mathbf{Y}_{t+1}^v$ is a matrix by concatenating matrices $\mathbf{Y}_t^v$ and $\sqrt{tk/\left(t+k\right)}\left(\overline{\mathbf{X}}_t^T - \mathbf{X}_{t+1}^T\right)$.

Equations (23) and (24) provides a way of quickly deducing projection matrices $\mathbf{U}_{t+1}$ and $\mathbf{V}_{t+1}$. Taking $\mathbf{U}_{t+1}$ for example, instead of directly calculating the eigenvalue decomposition of $\mathbf{Y}_{t+1}^u$ to find $\mathbf{U}_{t+1}$, we can apply the incremental SVD algorithms [27], [28] to calculate $\mathbf{U}_{t+1}$ by operating on $\mathbf{U}_t$ and $\sqrt{tk/\left(t+k\right)}\left(\overline{\mathbf{X}}_t - \mathbf{X}_{t+1}\right)$ with $\mathbf{U}_t$ obtained from the eigenvalue decomposition of $\mathbf{Y}_t^u$. Similarly, $\mathbf{V}_{t+1}$ can be deduced by operating on $\mathbf{V}_t$ and $\sqrt{tk/\left(t+k\right)}\left(\overline{\mathbf{X}}_t^T - \mathbf{X}_{t+1}^T\right)$.

Algorithm 3 shows the complete algorithm of our sequential anomaly detection. The quick deduction to calculate the projection matrices $\mathbf{U}_{t+1}$ and $\mathbf{V}_{t+1}$ is shown from line 2 to line 4 in Algorithm 3. On Eq.(31), $\mathbf{U}_{t+1}$ for $\mathbf{Y}_{t+1}^u$ is derived by reusing $\mathbf{U}_t$. As a result, compared with the direct method that finds $\mathbf{U}_{t+1}$ through the eigenvalue decomposition of $\mathbf{Y}_{t+1}^u$, the computation cost of our approach can be largely reduced. Similarly, we can determine $\mathbf{V}_{t+1}$.

With $\mathbf{M}_t = [\mathbf{U}_t, \mathbf{V}_t]$, $\mathbf{M}_{t+1} = [\mathbf{U}_{t+1}, \mathbf{V}_{t+1}]$, we can quickly detect the anomaly based on the metric in Eq. (9). As projection matrices should capture the basic features of the traffic data, in our sequential anomaly detection algorithm, we only update the projection matrices utilized for the next time slot when we detect the newly arriving data point is normal (line 9 - line 11).

Algorithm 3 Sequential anomaly detection algorithm

Input: The new sampling data matrix $\mathbf{X}_{t+1}$
The projection matrices of history data $\mathbf{U}_t$,$\mathbf{V}_t$
Eigen-values of history data $\mathbf{\Sigma}_t^u$,$\mathbf{\Sigma}_t^v$
Output: $\mathbf{U}_{t+1}$,$\mathbf{V}_{t+1}$
$\mathbf{\Sigma}_{t+1}^u$,$\mathbf{\Sigma}_{t+1}^v$
identify $\mathbf{X}_{t+1}$ is anomaly data

1: $\mathbf{B}^u = \sqrt{tk/\left(t+k\right)}\left(\overline{\mathbf{X}}_t - \mathbf{X}_{t+1}\right)$, $\mathbf{B}^v = \sqrt{tk/\left(t+k\right)}\left(\overline{\mathbf{X}}_t^T - \mathbf{X}_{t+1}^T\right)$

2: Processing QR decomposition for the following equation:

$$\mathbf{Q}^u\mathbf{R}^u = \left(I - \mathbf{U}_t(\mathbf{U}_t)^T\right)\mathbf{B}^u \tag{25}$$

$$\mathbf{Q}^v\mathbf{R}^v = \left(I - \mathbf{V}_t(\mathbf{V}_t)^T\right)\mathbf{B}^v \tag{26}$$

3: Processing SVD decomposition for the following equation

$$SVD\begin{bmatrix}\sqrt{\mathbf{\Sigma}_t^u} & (\mathbf{U}_t)^T\mathbf{B}^u \\ 0 & \mathbf{R}^u\end{bmatrix} = \tilde{\mathbf{U}}^u\tilde{\mathbf{\Sigma}}^u\left(\tilde{\mathbf{V}}^u\right)^T \tag{27}$$

$$SVD\begin{bmatrix}\sqrt{\mathbf{\Sigma}_t^v} & (\mathbf{V}_t)^T\mathbf{B}^v \\ 0 & \mathbf{R}^v\end{bmatrix} = \tilde{\mathbf{U}}^v\tilde{\mathbf{\Sigma}}^v\left(\tilde{\mathbf{V}}^v\right)^T \tag{28}$$

$$\mathbf{\Sigma}_{t+1}^u = \tilde{\mathbf{\Sigma}}^u \tag{29}$$

$$\mathbf{\Sigma}_{t+1}^v = \tilde{\mathbf{\Sigma}}^v \tag{30}$$

4: The updated projection matrices are computed as follows:

$$\mathbf{U}_{t+1} = [\mathbf{U}_t, \mathbf{Q}^u]\,\tilde{\mathbf{U}}^u \tag{31}$$

$$\mathbf{V}_{t+1} = [\mathbf{V}_t, \mathbf{Q}^v]\,\tilde{\mathbf{U}}^v \tag{32}$$

5: Build the combination projection matrices $\mathbf{M}_t = [\mathbf{U}_t, \mathbf{V}_t]$, $\mathbf{M}_{t+1} = [\mathbf{U}_{t+1}, \mathbf{V}_{t+1}]$

$$Cosine = \left|\frac{\langle Vec\left(\mathbf{M}_t\right), Vec\left(\mathbf{M}_{t+1}\right)\rangle}{\|Vec\left(\mathbf{M}_t\right)\|\,\|Vec\left(\mathbf{M}_{t+1}\right)\|}\right| \tag{33}$$

6: **if** $Cosine \geq score$ **then**
7: $\mathbf{X}_{t+1}$ is not anomaly data
8: **else**
9: $\mathbf{X}_{t+1}$ contains anomalous data
10: $\mathbf{U}_{t+1} = \mathbf{U}_t$,$\mathbf{V}_{t+1} = \mathbf{V}_t$
11: $\mathbf{\Sigma}_{t+1}^u = \mathbf{\Sigma}_t^u$,$\mathbf{\Sigma}_{t+1}^v = \mathbf{\Sigma}_t^v$
12: **end if**

VII. Computation Complexity Analysis

Based on B-PCA, three anomaly detection algorithms can be implemented. The first is our basic anomaly detection algorithm B-PCA (Algorithm 2), termed **IterBPCA**. When new data arrive, the monitoring data set is updated and then the iterative B-PCA is applied to calculate the principle

directions. The second is **ABPCA**. Different from **IterBPCA**, the principle directions are calculated using the approximate covariance matrices $\mathbf{C}_t^u$ and $\mathbf{C}_t^v$ in Section V. The third is our **OnlineBPCA** (Algorithm 3). Different from ABPCA, Algorithm 3 updates the principal directions based on the newly added monitoring data and the principal directions of history data. In the remaining of this section, we provide analyses on the computation cost of these three algorithms.

Let N be the number of nodes in the network and t be the total time slots monitored. IterBPCA involves multiple iterations. In each iteration, the projection matrix $\mathbf{U}$ is calculated and updated by first computing the covariance matrix and then the eigenvalue decomposition, which incurs the time complexity at $\mathrm{O}\left(N^3t^2\right)$ and $\mathrm{O}\left(N^3\right)$ respectively. Therefore, the time complexity of calculating the matrix $\mathbf{U}$ is $\mathrm{O}\left(N^3t^2\right)+\mathrm{O}\left(N^3\right)$. Similarly, we can derive that the time complexity of calculating the matrix $\mathbf{V}$ is also $\mathrm{O}\left(N^3t^2\right)+\mathrm{O}\left(N^3\right)$. With the total number of iteration rounds in IterBPCA denoted as L, the overall time complexity of IterBPCA can be expressed as $L\times(\mathrm{O}\left(N^3t^2\right)+\mathrm{O}\left(N^3\right))$.

ABPCA can directly calculate the projection matrices $\mathbf{U},\mathbf{V}$ through one time of covariance matrix computation and one time of eigenvalue decomposition, therefore the time complexity is $\mathrm{O}\left(N^3t^2\right)+\mathrm{O}\left(N^3\right)$.

OnlineBPCA is an incremental algorithm in which the projection matrices are updated with newly added data. To calculate the matrix $\mathbf{U}$, a QR decomposition (in (25)) and an eigenvalue decomposition (in (27)) are required, which involves the complexity of $\mathrm{O}\left(N^3\right)$, $\mathrm{O}\left((N+r)^3\right)$, respectively. Similarly, the time complexity of calculating the matrix $\mathbf{V}$ is $O\left(N^3\right)+\mathrm{O}\left((N+l)^3\right)$.

As t is the total number of time slots monitored, generally we have $N^3t^2>(N+r)^3$ and $N^3t^2>(N+l)^3$. Therefore, we have $Cost\left(IterBPCA\right)>Cost\left(ABPCA\right)>Cost\left(OnlineBPCA\right)$. In the simulation part, we also validate these results.

VIII. Performance Evaluations

Before we present the performance results, we first introduce our simulation setup.

A. Generation of Corrupted Synthesized Data

We synthetically generate anomalies by adding data outliers into the public traffic traces Abilene [23] and GÈANT [24]. As Abilene and GÈANT traffic traces record the volume of traffic flows between all source and destination pairs, they allow us to form a network-wide traffic matrix. Abilene and GÈANT collect monitoring data every 5 minutes and 15 minutes respectively, so the duration of a time slot for these two traces are different.

Following [29]–[31], we generate the corrupted data by selecting a set of OD flows from the raw data trace and setting them to be anomalous with the following strategies:

Step 1 (Data Normalization): For more efficient data processing, data normalization is often applied in the data preprocessing step to scale the variables or features of data.

We denote the raw trace data as $\mathcal{X}=\{\mathbf{X}_1,\mathbf{X}_2,\cdots,\mathbf{X}_k,\cdots,\mathbf{X}_t\}$ with $\mathbf{X}_k\in R^{N\times N}$, where N is the number of nodes in the network and t is the total number of time slots monitored. Given $X_k(i,j)$, we adopt the following equation to normalize the data:

$$X_k\left(i,j\right)=\frac{X_k\left(i,j\right)-\min\limits_{k,i,j}\left(X_k\left(i,j\right)\right)}{\max\limits_{k,i,j}\left(X_k\left(i,j\right)\right)-\min\limits_{k,i,j}\left(X_k\left(i,j\right)\right)} \tag{34}$$

where $\max\limits_{k,i,j}\left(X_k\left(i,j\right)\right)$ and $\min\limits_{k,i,j}\left(X_k\left(i,j\right)\right)$ are respectively the maximum value and minimum value of all the traffic data.

Step 2 (Anomaly Injection): As mentioned in Section I, traffic measurement data generally come in sequence. In the simulation, in each sequential step, we add into the data set the measurement sample from one more time slot. Then we apply our OnlineBPCA to the measurement data to detect whether this newly arriving data matrix contains anomalous samples or not.

We set the measurement data of the first week as training data, which are utilized in our model to capture the basic information of the traffic data. After the first week, we inject the outliers to the normalized traffic data to generate the corrupted data.

We randomly select $\gamma\times T$ time slots to inject the outlier data where T is the total time slots monitored except the first week. For each selected time slot, we randomly select $\beta\times(N\times N)$ positions to inject the anomalous data. We set $\beta=0.1$ and $\gamma=0.1$ as the default setting.

Outlier data are injected following the Gaussian distribution $\mathcal{N}\left(\mu,\sigma^2\right)$ with the mean μ and the variance σ^2. In this paper, we set $\mu=0.01$, $\sigma=0.01$ (Abilene) and $\mu=0.0001$, $\sigma=0.0001$ (GÈANT) as the default setting. To investigate how anomaly intensity impacts the detection performance, we vary σ^2, mean μ, and β of the injected outliers in Section VIII-D.

B. Performance Metric

We use the following three metrics to evaluate the performance of the implemented anomaly detection algorithms.

False Positive Rate (FPR): It measures the proportion of non-outliers that are wrongly identified as outliers.

True Positive Rate (TPR): It measures the proportion of outliers that are correctly identified.

In some literature studies, FPR and TPR are also called the false alarm rate and the detection accuracy, respectively. From the definitions, we can see that a smaller FPR or larger TPR value means a better performance.

Computation Time: It measures the average number of seconds taken to complete one sequence of anomaly detection steps.

All simulations are run on a Microstar workstation, which is equipped with two Intel (R) Xeon (R) E5-2620 CPUs with 2GHz processor, 24 Cores and 32 GB RAM. To evaluate the recovery computation time, we insert a timer to all approaches.

C. Parameter Setting and Impact

We first investigate the parameters used in the OnlineBPCA, based on which, we provide proper parameter setting for performance studies of OnlineBPCA in our simulations.

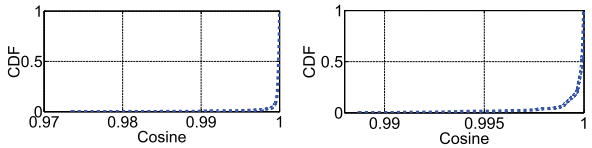


Fig. 8. Score with different λ. (a) Abilene. (b) GÈANT.

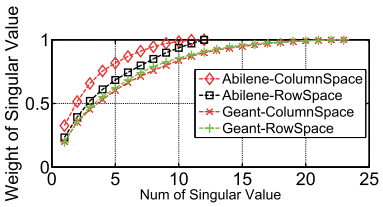


Fig. 9. The weight of different rank.

1) Score: According to our B-PCA based algorithm, if the projections of newly added data result in a large principal direction change and thus the Cosine is lower than the pre-defined score, we will claim that the data contain anomalous values. We set the score value empirically. Specifically, we calculate Cosine of each time step of the first week. From the probability theory and statistics, we know that the cumulative distribution function (CDF) describes the probability that a real random variable $X \leq x$. That is $F\ (x) = p\ (X \leq x)$. Fig. 8 shows the CDF results of Cosine in the first week. We do not inject anomalies into the traffic data of the first week to obtain the baseline traffic data without outliers. As nearly all the Cosine is larger than 0.98 (Abilene) and 0.99(GÈANT), we set score=0.98 and score=0.99 for trace data Abilene and GÈANT in the remaining tests. For other peer algorithms, we follow the same method to choose their threshold scores.

2) Rank: To detect anomaly, B-PCA transforms an $N \times N$ monitoring matrix to $r \times l$ projection matrix using left (column) projection matrix $\mathbf{U} \in R^{N \times r}$ and right (row) projection matrix $\mathbf{V} \in R^{N \times l}$. According to our approximate algorithm, $\mathbf{U} \in R^{N \times r}$ can be obtained by applying an eigenvalue decomposition of the covariance data matrix $\mathbf{C}_t^u$, i.e.

$$\mathbf{C}_t^u = \mathbf{Q}\Lambda\mathbf{Q}^T \tag{35}$$

where each column of $\mathbf{Q}$ represents an eigenvector of $\mathbf{C}_t^u$, and the corresponding diagonal entry in $\Lambda = diag(\lambda_1, \lambda_2, \cdots, \lambda_n)$ is the associated eigenvalue. For the purpose of dimension reduction, $\mathbf{U} = \mathbf{Q}_r$, where $\mathbf{Q}_r$ denotes the columns of $\mathbf{Q}$ associated with the largest r eigenvalues of $\mathbf{C}_t^u$. In order to keep as much original information as possible, the dimensionality of the subspace r is determined in terms of $\frac{\sum_{i=1}^{r}\lambda_i}{\sum_{i=1}^{n}\lambda_i} \geq \theta$ (where θ is the ratio of variation in the sub-space to the total variation in the original space, in general, $\theta = 98\%$). Similarly, we can also set l to preserve most variation of $\mathbf{C}_t^v$.

Fig.9 shows $\frac{\sum_{i=1}^{r}\lambda_i}{\sum_{i=1}^{n}\lambda_i}$ of covariance data matrices of the whole data in Abilene [23] and GÈANT [24]. According to the result, we set $r = 10$, $l = 11$ for the trace Abilene, $r = 20$, $l = 20$ for trace GÈANT, respectively.

3) Oversampling ratio: In Section VI-B, we propose a strengthening method to amplify the impact of newly arriving monitoring data on the principal directions of the monitoring data set. We duplicate the newly arriving data point k times

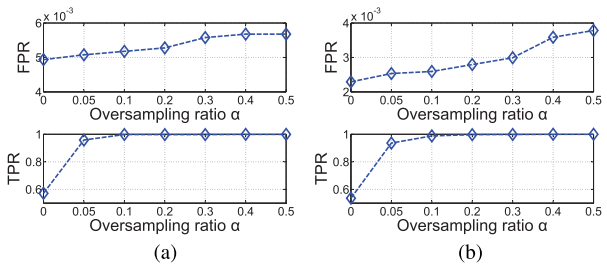


Fig. 10. The impact of sampling strengthen. (a) Abilene. (b) GÈANT.

while not changing the set of history data. In this paper, we set k as $k = \alpha \cdot t$, where t is the number of history data points and α is the oversampling ratio. We vary the oversampling ratio α to investigate how k impacts the performance of anomaly detection.

In Fig.10, the true positive rate(TPR) increases before α reaches $\alpha = 0.1$, beyond which the true positive rate is 1. Therefore, no more data duplication is needed after $\alpha = 0.1$. On the other hand, we notice that the false positive rate(FPR) also increases as α becomes larger, which is consistent with the tradeoff analysis on the impact of k in Section VI-B. Fortunately, the highest false positive rate is less than 0.6% for Abilene and 0.4% for GÈANT. Therefore, the oversampling ratio α impacts the FPR(false positive rate) very slightly. Accordingly, we set $\alpha = 0.1$ for both traffic traces.

For fairly comparing with other peer algorithms, we also apply the sample strengthening with α set following the above procedure to choose the smallest value that achieves the best anomaly detection performance.

D. Comparison with peer algorithms

As introduced in Section I, current traffic anomaly detection algorithms can be classified into two types. The first checks the variations of the statistical information to identify whether newly data contain anomalies. In the second type, anomalies in a period are detected through data separation. According to the two types, we have done two groups of simulations.

1) Detecting Anomaly Through Checking Variations: In the first group of simulations, by checking the variations of statistical information, we implement seven anomaly detection algorithms for performance comparison.

The first two are vector-based anomaly detection algorithms in [18], denoted as **VecPCA** and **IVecPCA**. Both algorithms model the monitoring data in a time slot as a vector and apply vector-based PCA to detect whether the newly arriving data point is an anomaly. Different from VecPCA, IVecPCA is an incremental algorithm which updates the first principal component with new data using a least squares approximation.

As introduced in Section I, the monitoring data in each time slot can be naturally modeled as a 2D matrix. To capture more features in the traffic data for more accurate anomaly detection, we propose to detect anomaly based on B-PCA, where the 2D data matrices are applied to construct the corresponding covariance matrix and the principal directions are calculated from both the column mode and the row mode. As we are not aware that any existing anomaly detection algorithm is designed based on 2D matrix model, for performance comparison, we implement five anomaly detection algorithms

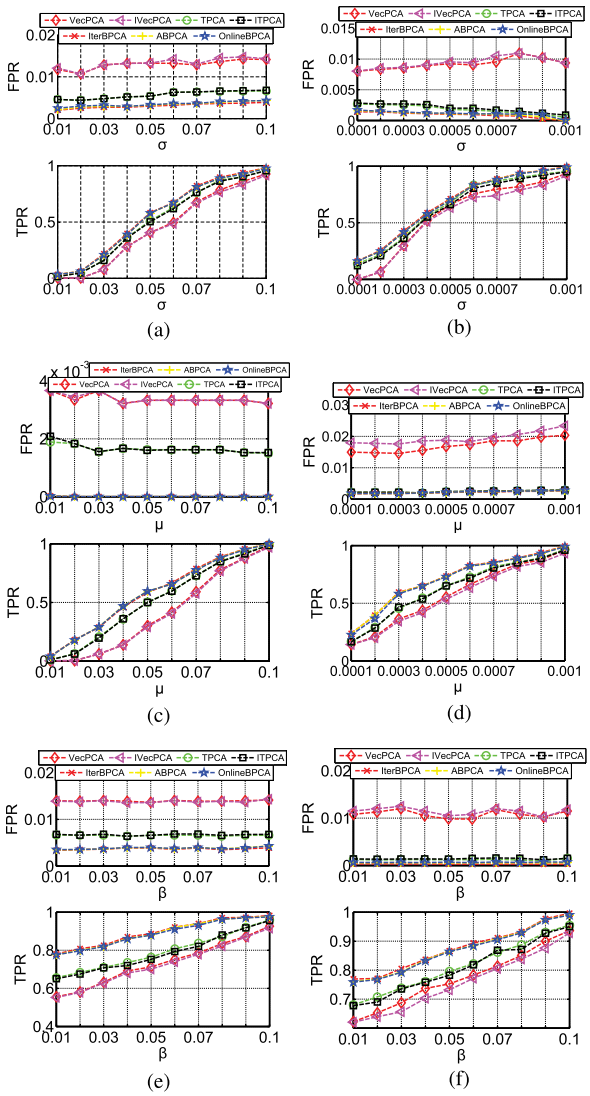


Fig. 11. Performance comparison with random anomalies (first group of simulations). (a) Abilene: random anomaly with different σ. (b) GÈANT: random anomaly with different σ. (c) Abilene: random anomaly with different μ. (d) GÈANT: random anomaly with different μ. (e) Abilene: random anomaly with different β. (f) GÈANT: random anomaly with different β.

based on PCA using 2D matrix model. The first three are based on B-PCA: **IterBPCA**, **ABPCA**, and **OnlineBPCA** (described in Section VII).

Besides B-PCA, there exists another PCA algorithm (2D-PCA [19]) that can deal with 2D matrix while only the principal direction of the column mode is derived. Although 2D-PCA is designed originally for image, for performance comparison, we also implement two anomaly detection algorithms **TPCA** and **ITPCA** based on 2D-PCA. TPCA directly applies 2D-PCA to calculate the principal directions for anomaly detection. Different from TPCA, in ITPCA, we apply the method proposed in Section VI-C to incrementally update the principal directions of 2D-PCA.

- *Accuracy Comparison*

To compare the detection accuracy of different anomaly detection algorithms, with other parameters fixed, we vary the variance σ^2, the average value μ, the outlier ratio β. Fig.11 shows the performance results under the random anomaly attack.

Among all the algorithms, our OnlineBPCA, ABPCA, IterBPCA achieve the best detection performance with the lowest false positive rate(FPR) and the highest true positive rate(TPR) under all simulation scenarios. For example, in Fig. 11(a), the false positive rates under our OnlineBPCA, ABPCA, IterBPCA are less than $1/3$ that under VecPCA and IVecPCA, and less than $2/3$ that under TPCA and ITPCA. These results demonstrate that our OnlineBPCA, ABPCA, IterBPCA can fully utilize the hidden features in the traffic data to accurately detect the anomaly and are robust to different kinds of anomalies.

As expected, TPCA and ITPCA achieve a better performance than VecPCA and IVecPCA because TPCA and ITPCA are designed based on 2D matrix data to capture more features for anomaly detection while VecPCA and IVecPCA are designed based-on vector data. Compared with our OnlineBPCA, ABPCA, IterBPCA, the performance under TPCA and ITPCA is worse as B-PCA can extract the information from both row side and column side to better detect anomalous data while 2D-PCA can only utilize information from one dimension.

Moreover, the similar performance under OnlineBPCA, ABPCA, IterBPCA demonstrates that our proposed approximation algorithm is very effective in approaching the performance of the sequential algorithm but at much lower computation complexity.

In the next section, we will further show that although OnlineBPCA, ABPCA, IterBPCA achieve similar detection accuracy, the computation time under OnlineBPCA is much smaller than that under ABPCA and IterBPCA even when the data set becomes larger with more monitoring data in the set.

As shown in Fig.11, with the increase of variance σ^2 and mean μ of the outliers, the True Positive Rate increases while the False Positive Rate almost do no change under all algorithms implemented, which implies that False Positive Rate is not sensitive to variance σ^2 and mean μ. Obviously, when the variance and mean of outliers are smaller, synthesized outlier data have closer and smaller values, and are more difficult to be differentiated from the normal data. Therefore, when variance σ^2 and mean μ are small, the True Positive Rate is small. As expected, in Fig.11(e) and Fig.11(f), with the increase of outlier injection ratio β thus the number of outliers, the True Positive Rate becomes larger under all algorithms.

- *The Computation Time*

In order to test the scalability of the proposed algorithm in handling the data set with the size increased over time, Table.II and Table.III shows the computation time required in one sequential anomaly detection for the time duration of 100 time slots after the first week.

Obviously, the history data set becomes larger as time slot number increases. The three incremental algorithms IVecPCA, ITPCA, OnlineBPCA achieve the stable and smallest computation cost over the time, while the computation cost under other algorithms (VecPCA, TPCA, ABPCA, InterBPCA) increases over the time. These performance results demonstrate that our incremental algorithms have good scalability. Compared with IterBPCA, the computation time under ABPCA is largely reduced as ABPCA calculates principal directions through

TABLE I

COMPUTATION TIME OF DETECT ALGORITHMS WHEN DATA SET BECOMES LARGE AT THE 5000TH TIME SLOT

Time(sec.)	IVecPCA	VecPCA	TPCA	ITPCA	IterBPCA	ABPCA	OnlineBPCA
Abilene	1.8E-4	0.628	176.36	$2.0E-4$	$2.0E+4$	178.31	$2.0E-4$
GEANT	$1.4E-3$	1.983	708.47	$2.4E-3$	$8.0E+4$	721.26	$2.5E-3$

TABLE II

COMPUTATION TIME COMPARISON UNDER *Abilene*

Time slot after first week	20	40	60	80	100
IVecPCA	0.00022	0.00023	0.00023	0.00023	0.00023
VecPCA	0.0252	0.04732	0.06723	0.08563	0.1321
TPCA	0.2022	0.2636	0.3883	0.4732	0.6052
ITPCA	0.00032	0.00033	0.00033	0.00032	0.00033
IterBPCA	2.7372	3.663	4.7688	6.2897	6.9357
ABPCA	0.2335	0.3367	0.4367	0.5836	0.7573
OnlineBPCA	0.00043	0.00041	0.00041	0.00043	0.00043

TABLE III

COMPUTATION TIME COMPARISON UNDER *GEANT*

Time slot after first week	20	40	60	80	100
IVecPCA	0.00052	0.00053	0.00053	0.00053	0.00053
VecPCA	0.3952	0.4778	0.5873	0.7568	0.9381
TPCA	1.2615	1.6985	2.1279	2.8317	3.5219
ITPCA	0.00052	0.00051	0.00051	0.00053	0.00051
IterBPCA	12.3379	15.9918	20.0780	25.6787	30.6717
ABPCA	1.2723	1.7393	2.3933	3.0971	3.7563
OnlineBPCA	0.00097	0.00097	0.00097	0.00097	0.00097

the approximation while IterBPCA calculates the directions through costly iterations. Rather than processing all the past data together, our proposed OnlineBPCA performs online anomaly detection within each time slot, and the detection time does not increase as time slot number increases.

Table I further shows the computation time for one sequential anomaly detection when the history data set becomes very large at the 5000th time slot. Among IterBPCA, ABPCA, and OnlineBPCA that achieve similar detection accuracy as shown in Fig.11, our OnlineBPCA requires the least time $2.0E-4$ seconds, while the computation time under IterBPCA and ABPCA increases and reaches $2.0E+4$ seconds and 178.31 seconds, respectively. This demonstrates that our approximation algorithm and sequential update of the principal direction can bring in a significant computation cost reduction. Compared with IterBPCA, the computation time under ABPCA is largely reduced as ABPCA calculates principal directions through the approximation while IterBPCA calculates the directions through costly iterations.

Although IVecPCA and ITPCA achieve the similar computation cost as our OnlineBPCA, from Fig.11, we can find that the detection accuracy of these two algorithms are much worse than our OnlineBPCA.

All the simulation results demonstrate that our OnlineBPCA is a robust anomaly detection algorithm with good scalability, and can quickly and accurately detect the anomaly data even when the history data set is very large.

2) Identifying anomaly through data separation: Besides our OnlineBPCA, we implement four data separation based anomaly detection algorithms (DRMF [17], RPCA [14], PCA [6], and Bilinear Factor Matrix Norm Minimization (BFMNM) [16]) for performance comparison. The work

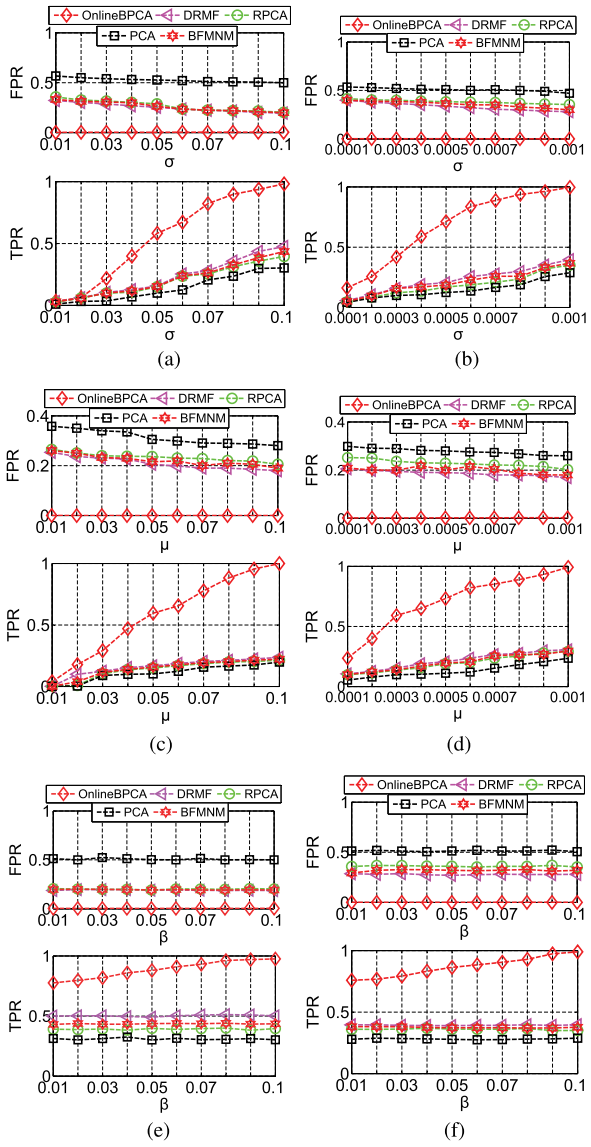


Fig. 12. Performance comparison with random anomalies (second group of simulations). Abilene: random anomaly with different σ. (b) GÈANT: random anomaly with different σ. (c) Abilene: random anomaly with different μ. (d) GÈANT: random anomaly with different μ. (e) Abilene: random anomaly with different β. (f) GÈANT: random anomaly with different β.

in [16] proposes two BFMNM models, using l_p-norm minimization with $p = 1/2$ and $p = 2/3$ respectively. Experimental results show that the two models behave similarly, and the model using $p = 2/3$ performs slightly better. Thus we implement BFMNM with $p = 2/3$ in this paper as performance reference.

Following most of traffic anomaly detection algorithms, these four algorithms use the traffic matrix model in Fig.1. As DRMF, RPCA, PCA, and BFMNM aim to find the anomalies in a period rather than only in the new data, to fairly compare with these algorithms, we adopt the following detection

TABLE IV
COMPUTATION TIME COMPARISON UNDER *Abilene*

Time slot after first week	20	40	60	80	100
OnlineBPCA	0.00043	0.00041	0.00041	0.00043	0.00043
DRMF	8.1992	10.0322	12.7387	15.9829	20.8906
RPCA	2.2237	4.7037	6.0906	7.0712	10.3980
PCA	0.2529	0.3593	0.4311	0.5367	0.6862
BFMNM	0.7852	1.5896	2.6856	3.2568	3.98765

TABLE V
COMPUTATION TIME COMPARISON UNDER *GEANT*

Time slot after first week	20	40	60	80	100
OnlineBPCA	0.00097	0.00097	0.00097	0.00097	0.00097
DRMF	38.9890	51.7618	57.6219	67.3318	81.0619
RPCA	20.6501	27.3926	35.1270	41.3250	52.3292
PCA	1.37667	1.8329	2.3518	2.9976	3.7116
BFMNM	8.3568	9.53568	12.3568	15.3658	18.6589

principle: among all the candidate anomaly locations, return the η locations with the largest η absolute values where η is the number of anomalies injected. If the newly arriving data contain anomalies in any locations, the newly arriving data is identified to be anomalous; otherwise, the newly arriving data is considered normal.

- *Accuracy Comparison*

We plot the simulation results under random anomalies in Fig.12. As expected, compared with DRMF, RPCA, PCA, and BFMNM, our OnlineBPCA achieves the best performance with the lowest False Positive Rate and the highest True Positive Rate under all the simulation scenarios. Although DRMF, RPCA, PCA, and BFMNM are effective in separating normal and anomalous data, they are not good at identifying whether newly data contain anomalies for on-line anomaly detection as any false data separation in the history will impact the detection performance for the new data.

Moreover, consistent with the literature studies [1], [17], among DRMF, RPCA, PCA, and BFMNM, the best performance is achieved under DRMF as it formulates the problem directly using the matrix rank to represent the low rank feature and the L_0-norm to represent the sparse feature of the anomalous data.

- *The Computation Time*

Obviously, the history data set becomes larger as time slot number increases. In Tables IV and V, our algorithm OnlineBPCA achieves the stable and smallest computation cost throughout the test time, while the computation cost under other algorithms (DRMF, RPCA, PCA, and BFMNM) increases over the time.

As DRMF, RPCA, and BFMNM separate the observed traffic matrix through costly iterations, their computation time is very large. Compared with DRMF, RPCA, and BFMNM, the computation time under PCA is much lower, and the time of our OnlineBPCA is several orders lower than all schemes studied. Consistent with the literature study [16], compared with RPCA, BFMNM achieves higher anomaly detection accuracy with much smaller computation time. Although both PCA and OnlineBPCA require SVD operations, PCA operates on the whole traffic data of a period, while OnlineBPCA can reuse the matrix decomposition of the previous step to deduce the decomposition of current data. Thus computation cost can be largely reduced, and we can conclude that our OnlineBPCA can be applied to quickly identify sudden anomalies in real time.

IX. CONCLUSION

We propose a novel anomaly detection algorithm, OnlineBPCA, the first anomaly detection algorithm based on two-dimensional PCA. We propose several novel techniques in OnlineBPCA to support quick and accurate anomaly detections, which include 1) a novel anomaly detection principle which enables the application of B-PCA for anomaly detection, 2) an approximate algorithm to avoid using the iteration procedure to calculate the principal directions, 3) a sequential anomaly detection algorithm that does not require the storage of the past data and can update the principal directions using the current monitoring data, and 4) a strengthening method to amplify the impact of newly arriving monitoring data for more accurately detecting anomalies in the new data when the historical data become large. Using real traffic traces, we have done extensive simulations to compare our OnlineBPCA with the state of art anomaly detection algorithms. Our simulation results demonstrate that, compared with other algorithms, our OnlineBPCA can achieve significantly better detection performance with low False Positive Rate, high True Positive Rate, and very low computational cost.

REFERENCES

[1] G. Xie *et al.*, "Fast low-rank matrix approximation with locality sensitive hashing for quick anomaly detection," in *Proc. IEEE INFOCOM*, May 2017, pp. 1–9.

[2] K. Xie *et al.*, "Fast tensor factorization for accurate Internet anomaly detection," *IEEE/ACM Trans. Netw.*, vol. 25, no. 6, pp. 3794–3807, Dec. 2017, doi: 10.1109/TNET.2017.2761704.

[3] H. Huang, H. Al-Azzawi, and H. Brani. (Feb. 2014). "Network traffic anomaly detection." [Online]. Available: https://arxiv.org/abs/1402.0856

[4] D. Jiang, Z. Xu, P. Zhang, and T. Zhu, "A transform domain-based anomaly detection approach to network-wide traffic," *J. Netw. Comput. Appl.*, vol. 40, no. 2, pp. 292–306, Apr. 2014.

[5] A. Lakhina *et al.*, "Structural analysis of network traffic flows," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 32, no. 1, pp. 61–72, 2004.

[6] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *Proc. Conf. Appl., Technol., Archit., Protocols Comput. Commun. (SIGCOMM)*, vol. 34. Oct. 2004, pp. 219–230.

[7] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *J. Educ. Psychol.*, vol. 24, no. 6, p. 417, 1933.

[8] A. Lakhina, M. Crovella, and C. Diot, "Characterization of network-wide anomalies in traffic flows," in *Proc. 4th ACM SIGCOMM Conf. Internet Meas.*, 2004, pp. 201–206.

[9] L. Huang *et al.*, "In-network PCA and anomaly detection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 617–624.

[10] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 4, pp. 217–228, 2005.

[11] L. Huang *et al.*, "Communication-efficient online detection of network-wide anomalies," in *Proc. 26th IEEE Int. Conf. Comput. Commun. (INFOCOM)*, May 2007, pp. 134–142.

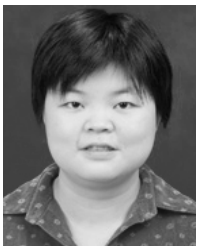
[12] C. Callegari, L. Gazzarrini, S. Giordano, M. Pagano, and T. Pepe, "A novel pca-based network anomaly detection," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2011, pp. 1–5.

[13] Z. Lin, M. Chen, and Y. Ma. (Sep. 2010). "The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices." [Online]. Available: https://arxiv.org/abs/1009.5055

[14] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3, p. 11, May 2011.

[15] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[16] F. Shang, J. Cheng, Y. Liu, Z.-Q. Luo, and Z. Lin, "Bilinear factor matrix norm minimization for robust PCA: Algorithms and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published, doi: : 10.1109/TPAMI.2017.2748590.
[17] L. Xiong, X. Chen, and J. Schneider, "Direct robust matrix factorizatoin for anomaly detection," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2011, pp. 844–853.
[18] Y. J. Lee, Y. R. Yeh, and Y. C. F. Wang, "Anomaly detection via online oversampling principal component analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 7, pp. 1460–1470, Jul. 2013.
[19] J. Yang, D. Zhang, A. F. Frangi, and J.-Y. Yang, "Two-dimensional PCA: A new approach to appearance-based face representation and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 1, pp. 131–137, Jan. 2004.
[20] H. Kong *et al.*, "Generalized 2D principal component analysis for face image representation and recognition," *Neural Netw.*, vol. 18, nos. 5–6, pp. 585–594, 2005.
[21] H. Yu and M. Bennamoun, "1D-PCA, 2D-PCA to nD-PCA," in *Proc. IEEE 18th Int. Conf. Pattern Recognit. (ICPR)*, vol. 4. Aug. 2006, pp. 181–184.
[22] H. Kong *et al.*, "Generalized 2D principal component analysis," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 1, Jul. 2005, pp. 108–113.
[23] *The Abilene Observatory Data Collections*. Accessed: Jul. 20, 2004. [Online]. Available: http://abilene.internet2.edu/observatory/data-collections.html
[24] S. Uhlig, B. Quoitin, J. Lepropre, and S. Balon, "Providing public intradomain traffic matrices to the research community," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 83–86, 2006.
[25] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, p. 15, 2009.
[26] D. Zhang and Z.-H. Zhou, "$(2D)^2$ PCA: Two-directional two-dimensional pca for efficient face representation and recognition," *Neurocomputing*, vol. 69, nos. 1–3, pp. 224–231, 2005.
[27] 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.
[28] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vis.*, vol. 77, nos. 1–3, pp. 125–141, 2008.
[29] B. I. P. Rubinstein *et al.*, "Compromising PCA-based anomaly detectors for network-wide traffic," Dept. Elect. Eng. Comput. Sci., Univ. California, Berkeley, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2008-73, 2008.
[30] R. A. Maxion and K. M. C. Tan, "Benchmarking anomaly-based detection systems," in *Proc. IEEE Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2000, pp. 623–630.
[31] A. Soule, K. Salamatian, and N. Taft, "Combining filtering and statistical methods for anomaly detection," in *Proc. 5th ACM SIGCOMM Conf. Internet Meas.*, 2005, p. 31.
[32] B. Zhang, J. Yang, J. Wu, D. Qin, and L. Gao, "PCA-subspace method—Is it good enough for network-wide anomaly detection," in *Proc. IEEE Netw. Oper. Manage. Symp. (NOMS)*, Apr. 2012, pp. 359–367.



Kun Xie received the Ph.D. degree in computer application from Hunan University, Changsha, China, in 2007. She is currently a Professor with Hunan University. Her research interests include network monitoring and management, wireless network, mobile computing, and matrix and tensor factorization.

Xiaocan Li is currently pursuing the Ph.D. degree with the College of Computer Science and Electronics Engineering, Hunan University, Changsha, China.

Xin Wang (M'01) received the Ph.D. degree in electrical and computer engineering from Columbia University, New York, NY, USA. She is currently an Associate Professor with the Department of Electrical and Computer Engineering, The State University of New York at Stony Brook, Stony Brook, NY, USA. Her research interests include algorithm and protocol design in wireless networks and communications, mobile and distributed computing, and networked sensing and detection. She received the NSF CAREER Award in 2005 and the ONR Challenge Award in 2010.

Jiannong Cao (M'93–SM'05–F'14) received the Ph.D. degree in computer science from Washington State University, Pullman, WA, USA, in 1990. He is currently a Chair Professor and the Head of the Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong. His research interests include parallel and distributed computing, computer networks, mobile and pervasive computing, fault tolerance, and middleware.

Gaogang Xie received the B.S. degree in physics and the M.S. and Ph.D. degrees in computer science from Hunan University, in 1996, 1999, and 2002, respectively. He is currently a Professor and the Director of the Network Technology Research Center, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His research interests include Internet architecture, packet processing and forwarding, and Internet measurement.

Jigang Wen received the Ph.D. degree in computer application from Hunan University, China, in 2011. He was a Research Assistant with the Department of Computing, Hong Kong Polytechnic University, from 2008 to 2010. He is currently a Post-Doctoral Fellow with the Institute of Computing Technology, Chinese Academy of Science, China. His research interests include wireless network and mobile computing and high-speed network measurement and management.

Dafang Zhang received the Ph.D. degree in application mathematics from Hunan University, Changsha, China, in 1997. He is currently a Professor with Hunan University. His research interests include packet processing, Internet measurement, wireless network and mobile computing, and big data.

Zheng Qin received the Ph.D. degree in computer science from Chongqing University, China, in 2001. He is currently a Professor with the College of Computer Science and Electronic Engineering, Hunan University, China. His main research interests include computer networking, network and information security, big data, and cloud computing.