Streaming Anomaly Detection Using Randomized Matrix Sketching

Hao Huang
General Electric Global Research
San Ramon, CA 94583
haohuanghw@gmail.com

Shiva Prasad Kasiviswanathan
Samsung Research America
Mountain View, CA 95134
kasivisw@gmail.com

ABSTRACT

Data is continuously being generated from sources such as machines, network traffic, application logs, etc. Timely and accurate detection of anomalies in massive data streams has important applications such as in preventing machine failures, intrusion detection, and dynamic load balancing. In this paper, we introduce a novel (unsupervised) anomaly detection framework which can be used to detect anomalies in a streaming fashion by making only one pass over the data while utilizing limited storage. We adapt ideas from matrix sketching to maintain, in a streaming model, a set of few orthogonal vectors that form a good approximate basis for all the observed data. Using this constructed orthogonal basis, anomalies in new incoming data are detected based on a simple reconstruction error test. We theoretically prove that our algorithm compares favorably with an offline approach based on expensive global singular value decomposition (SVD) updates. Additionally, we apply ideas from randomized low-rank matrix approximations to further speedup the algorithm. The experimental results show the effectiveness and efficiency of our approach over other popular scalable anomaly detection approaches.

1. INTRODUCTION

Detecting anomalies in huge volumes of data has many important real-life applications in areas such as machine health monitoring, intrusion detection systems, financial fraud detection, and medical diagnosis [9, 1]. However, it is also a challenging problem because in many modern applications the data arrives in a streaming fashion. The streaming data could be infinite, so offline algorithms that attempt to store the entire stream for analysis will not scale. Also in these situations, there is usually a lack of a complete (labeled) training set as new anomalous and non-anomalous patterns arise over time (this is sometimes referred to as *concept drift*). Another common requirement in many mission-critical applications is to detect anomalies in near real-time, as new data values are encountered. In this paper, we introduce novel approaches to anomaly detection in an unsupervised setting based on ideas from matrix sketching.

Although a lot of recent research has been focused on streaming anomaly detection [9, 1], there is still lack of theoretically sound and practically effective algorithms that operate efficiently by making just *one pass* over the data, which is an essential requirement for any "true" streaming algorithm. In practice, however, because of inherent correlations in the data, it is possible to reduce a large sized numerical stream into just a handful of hidden basis that can compactly describe the key patterns [35], and thereby dramatically reducing the complexity of further analysis. We exploit this observation in our proposed algorithms by maintaining a set of few orthogonal vectors that conceptually constitute up-to-date normal patterns.

A class of popular techniques for unsupervised anomaly detection, which can be referred to as *subspace-based* anomaly detection, operate by first constructing some sort of low-rank (e.g. principal component) matrix approximation of the input and then the projection of a new datapoint onto this low-rank matrix is used for deciding whether the point is anomalous or not [25, 23, 22]. Now this general idea can be utilized to construct a simple anomaly detection framework in a streaming setting: At time t, let us assume that we have a low-rank matrix U that can linearly represent well all the identified non-anomalous datapoints till time $t-1$. For a new datapoint $\mathbf{y}$ arriving at time t, if there does not exist a "good" representation[1] of $\mathbf{y}$ using U, then $\mathbf{y}$ does not lie close to the space of non-anomalous datapoints, and $\mathbf{y}$ could be an anomaly. After identifying the non-anomalous points, the low-rank matrix is updated to capture the insights from these non-anomalous points. Standard spectral theory informs that a straightforward way of maintaining such a low-rank matrix is to use repeated singular value decompositions on the whole observed dataset, as new non-anomalous data are identified. However, this is both computationally and storage intensive. Ideally, we want to maintain this low-rank matrix within a steaming setup (i.e., at any timepoint only the current non-anomalous datapoints are used to update the old low-rank matrix).

In this paper, we use *streaming matrix sketching*, to efficiently store and update a low-rank matrix (with orthogonal columns) that can linearly represent well over time the identified non-anomalous datapoints. Informally, a sketch of a matrix Z is another matrix Z' that is of smaller size than Z, but still approximates it well. For matrix sketching, we build upon and improve an elegant idea which was recently proposed by Liberty [28]. The matrix sketching algorithm in [28] (referred to as *Frequent Directions*) operates in a streaming model, accepts one datapoint at a time, and constructs a sketch matrix using a (surprisingly) simple idea of "shrinking" a few orthogonal vectors.

Our Contributions. We propose two streaming anomaly detection approaches operating in the above discussed framework. The approaches differ in the techniques used for matrix sketching. Since in our problem setting, more than one point could arrive at each timestep, our first approach is based on extending

[1] This can be equivalently stated as that projection of $\mathbf{y}$ onto the residual space of U is "big".

Proceedings of the VLDB Endowment, Vol. 9, No. 3
Copyright 2015 VLDB Endowment 2150-8097/15/11.

Frequent Directions [28] to efficiently deal with this scenario. Our algorithm for matrix sketching achieves a speedup over Frequent Directions that is almost linear in the number of datapoints handled at each timestep. Our second approach further improves this computational efficiency, by combining sketching with ideas from the theory of randomized low-rank matrix approximations. The computational efficiency gains of the (second) randomized approach over the (first) deterministic approach come at a cost of a small loss in the sketching performance.

We present a theoretical analysis of both our approaches to show that, under some reasonable assumptions, our approaches attain almost the same performance as a global approach that uses the entire data history at every timestep. The latter requires repeated and costly singular value decompositions over an ever increasing sized data matrix, while our proposed algorithms operate in a true streaming setting utilizing limited storage. We obtain these results by generalizing the analysis of Frequent Directions from [28, 14, 13] and by carefully combining it with recent spectral results in matrix perturbation and randomized low-rank matrix approximation theories. Our proposed anomaly detection algorithms have the following salient features:

(1) They can identify anomalies in close to real time, ensuring that the detection keeps up with the rate of data collection.

(2) They are pass-efficient, in that only one pass is required for each datapoint.

(3) They are space-efficient and require only a small amount of bookkeeping space.

(4) They operate in an unsupervised setting, but regular model updates allow them to still easily adapt to unseen normal patterns (concept drift) in the data.

Our experimental results corroborate the excellent performance of our approaches on datasets drawn from diverse domains such as biomedical, network security, and broadcast news, even in presence of concept drifts. Additionally, our algorithms are significantly more time and space efficient, compared to other popular scalable anomaly detection algorithms.

2. RELATED WORK

Anomaly detection is a well-studied topic and we refer the reader to the excellent surveys by Chandola *et al.* [9] and Aggarwal [1] for an introduction. We mention a few relevant researches here.

Many anomaly detection approaches have been suggested based on approximating the sample density. This includes the distance-based methods [5] and the manifold based methods [19, 20, 21]. However, these methods do not work well on large datasets since they require either computing all pair-wise distances or the complete affinity matrix, both of which are time and space consuming. Recently, inlier-based outlier detection methods were proposed in [18]. However, their training and computational complexity requirements render them unsuitable for real-time streaming applications.

There are many (semi-)supervised techniques that have been proposed for anomaly detection (refer to the surveys [1, 9]). They typically operate by finding a normal region containing a certain fraction of non-anomalous training samples; points outside the normal region are regarded as anomalies. These methods are faster than classic density measurement methods but their training requirements make them also unsuitable for real-time streaming applications.

Some other, more efficient techniques such as IForest [30] and Mass [36] are based on attribute-wise analysis. But they tend to fail when data has high dimensions or the distribution for anomalous points becomes less discriminative, e.g., if the anomalous and non-anomalous points share similar attribute range or distribution [21].

In streaming setup the training set is usually never perfect, and the detection model needs to be updated as new data comes in. The ideal scenario is to detect the arrival of a new normal pattern, and then improve the model suitably. Some methods achieve this by relying on probabilistic modeling of the data distributions and monitoring the likelihood for new-coming observations; see the survey by [33]. But they usually require accessing the whole of the past historical data at each timestep. Hence, these approaches are not practical for big data applications.

Kernel-based online anomaly detection algorithm proposed by [2] uses a dictionary learned over normal data to detect anomalies, but the high computation cost because of its growing dictionary strategy renders it unsuitable for large datasets.

Several techniques have been specifically designed for detecting outliers in time-series based data (see [31] and references therein). However, the techniques proposed in these studies, seem quite different to ideas presented here.

Most relevant to our work are the anomaly detection algorithms which are roughly based on identifying and tracking the dominant low-rank subspace of the data [25, 23, 22]. In Huang *et al.* [23, 22] anomaly detection decisions are made using a sliding time window. However, their algorithm requires costly PCA computation at each timestep, and is not practical for large window sizes. Even though we also use low-rank subspace tracking ideas, because of efficient sketching, our algorithms are highly scalable even while utilizing the entire data history.

Gabel *et al.* [11] recently proposed an anomaly detection scheme that uses a Johnson-Lindenstrauss style random subspace embedding to project each point to a lower dimensional space. Our algorithms, on the other hand, operate in the original feature space thereby avoiding the instability issues arising from random subspace embedding.

Our algorithms maintain (over time) a low-rank approximation to the input using matrix sketching. This is related to a line of work, referred to as Incremental Principal Component Analysis [17, 26, 8, 3, 4], where the goal is also to maintain a low-rank approximation of a matrix (using SVD and a small amount of bookkeeping) as rows/columns of a matrix arrive in a stream. However, most of these approaches lack rigorous theoretical guarantees on the quality of the maintained low-rank approximation. In fact, it has been shown that they can have arbitrarily bad matrix approximation error on adversarial data [14] and could suffer from practically poor performances [12]. Comparably, our algorithms have strong worst-case theoretical bounds and also have good empirical performance.

3. PRELIMINARIES

Notation. We denote $[n] = 1 : n$. Vectors are always in column-wise fashion and are denoted by boldface letters. For a vector $\mathbf{v}$, $\mathbf{v}^\top$ denotes its transpose and $\|\mathbf{v}\|$ denotes its Euclidean norm. For a matrix $Z \in \mathbb{R}^{m \times n} = \{z_{ij}\}$, its Frobenius norm $\|Z\|_F^2 = \sum_{ij} z_{ij}^2$, and its spectral norm $\|Z\| = \sup\{\|Z\mathbf{v}\| : \|\mathbf{v}\| = 1\}$. We use $\text{rank}(Z)$ to denote the rank and $\text{tr}(Z)$ to denote the trace of Z. We use $Z \succeq 0$ to denote that if Z is a positive semidefinite matrix, and if $Z - Y \succeq 0$, then we write $Z \succeq Y$. For a vector $(z_1, \dots, z_m) \in \mathbb{R}^m$, let $\text{diag}(z_1, \dots, z_m) \in \mathbb{R}^{m \times m}$ denote a diagonal matrix with $z_1, \dots, z_m$ as its diagonal entries. Given a matrix Z, we abuse notation and use $\mathbf{y} \in Z$ to represent that $\mathbf{y}$ is a column in Z. Let $\mathbf{I}_m$ denote an identity matrix of dimension $m \times m$. Given a set of matrices, $Z_1, \dots, Z_t \in \mathbb{R}^{m \times n_i}$, we use the notation $Z_{[t]} \in \mathbb{R}^{m \times n_{[t]}}$ where $n_{[t]} = \sum_{i=1}^t n_i$ to denote the matrix obtained by horizontally concatenating $Z_1, \dots, Z_t$, i.e., $Z_{[t]} = [Z_1, \dots, Z_t]$.

We use $\textsc{Svd}(Z)$ to denote the singular value decomposition of Z, i.e., $\textsc{Svd}(Z) = U\Sigma V^\top$. Here U is an $m\times m$ orthogonal matrix, Σ is an $m \times n$ diagonal matrix, and V is an $n \times n$ orthogonal matrix. The diagonal entries of Σ are known as the singular values of Z. Let $\sigma_i(Z)$ denote the ith singular value of Z. We follow the common convention to list the singular values in non-increasing order. For a symmetric matrix $S \in \mathbb{R}^{m\times m}$, we use $\textsc{Eig}(S)$ to denote its eigenvalue decomposition, i.e., $U\Lambda U^\top = \textsc{Eig}(S)$. Here U is an $m\times m$ orthogonal matrix and Λ is an $m\times m$ diagonal matrix whose (real) entries, $\lambda_1,\dots,\lambda_m$, are known as the eigenvalues of S (again listed in non-increasing order).

The best rank-k approximation (in both the spectral and Frobenius norm) to a matrix $Z \in \mathbb{R}^{m\times n}$ is $Z_k = \sum_{i=1}^k \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$, where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k$ are the top-k singular values of Z, with associated left and right singular vectors $\mathbf{u}_i \in \mathbb{R}^m$ and $\mathbf{v}_i \in \mathbb{R}^n$, respectively. We use $\textsc{Svd}_k(Z)$ to denote the singular value decomposition of Z_k, i.e., $Z_k = \textsc{Svd}_k(Z) = U_k\Sigma_k V_k^\top$. Here $\Sigma_k = \operatorname{diag}(\sigma_1,\dots,\sigma_k) \in \mathbb{R}^{k\times k}$, $U_k = [\mathbf{u}_1,\dots,\mathbf{u}_k] \in \mathbb{R}^{m\times k}$, and $V_k = [\mathbf{v}_1,\dots,\mathbf{v}_k] \in \mathbb{R}^{n\times k}$. The following celebrated theorem bounds the approximation error.

Theorem 1. *[15] Let $Z \in \mathbb{R}^{m\times n}$ with $n > m$, and let $\sigma_1 \geq \cdots \geq \sigma_m$ be the singular values of Z. Let $U_k\Sigma_k V_k^\top = \textsc{Svd}_k(Z)$. Then*

$$\min_{\operatorname{rank}(X)\leq k} \|Z - X\|_2 = \|Z - U_k\Sigma_k V_k^\top\|_2 = \sigma_{k+1}\,,$$

$$\min_{\operatorname{rank}(X)\leq k} \|Z - X\|_F = \|Z - U_k\Sigma_k V_k^\top\|_F = \sqrt{\sum_{j=k+1}^{m} \sigma_{k+1}^2}.$$

In this paper, Z_k always denotes the rank-k approximation of a matrix Z according to Theorem 1.

Definition 1. *Define the k-condition number of a matrix $Z \in \mathbb{R}^{m\times n}$ with $n > m$ as $\kappa_k(Z) = \sigma_1/\sigma_k \geq 1$ where $\sigma_1 \geq \cdots \geq \sigma_m$ are the singular values of Z.*

Our analysis uses the following simple (well-known) claim.

Claim 2. *Let $Z \in \mathbb{R}^{m\times n}$, and let Z_k be a rank-k approximation of Z according to Theorem 1. For any vector $\mathbf{x} \in \mathbb{R}^m$, $\kappa_k(Z)\|Z_k^\top \mathbf{x}\| \geq \|Z^\top \mathbf{x}\|$.*

4. STREAMING ANOMALY DETECTION

In this section, we propose an anomaly detection scheme for streaming data based on matrix sketching, and also provide theoretical guarantees for its efficacy. We start by describing the problem of streaming anomaly detection.

4.1 Problem Setting and Framework Overview

Streaming Anomaly Detection Task. We assume that the data arrives in streams and each datapoint has a timestamp that indicates when it arrives. The timestamp could be at any granularity, e.g., it could be the day or the exact time the datapoint arrives. Let $\{Y_t \in \mathbb{R}^{m\times n_t}, t = 1, 2, \dots\}$ denote a sequence of streaming data matrices, where Y_t represents the datapoints arriving at time t. Here m is the size of the feature space, and $n_t \geq 1$[2] is the number of datapoints arriving at time t. We normalize Y_t such that each column (point) in Y_t has a unit L_2-norm. Under this setup, the goal of streaming anomaly detection is to **identify "anomalous datapoints" in $\mathbf{Y_t}$ at every time $\mathbf{t}$.**

Our Anomaly Detection Framework. Our idea is based on maintaining, at every time t, a low-rank matrix with orthogonal columns that can linearly reconstruct "well" the entire prior (till time $t-1$) non-anomalous datapoints that the algorithm has identified. In other words, we identify a small set of (orthogonal) basis vectors that can represent well all the prior non-anomalous datapoints. At time t, a new point $\mathbf{y}_i \in Y_t$ is marked as anomalous if it cannot be well linearly reconstructed using these basis vectors (i.e., $\mathbf{y}_i$ does not lie "close" to the space of non-anomalous points).

Let $N_{[t-1]} = [N_1,\dots,N_{t-1}]$ be the set of all datapoints (columns) in $Y_{[t-1]} = [Y_1,\dots,Y_{t-1}]$ that the algorithm has identified as non-anomalous, with N_i denoting the set of datapoints in Y_i identified as non-anomalous. Consider the rank-k approximation of $N_{[t-1]}$ (for an appropriately chosen parameter k).[3]

$$N_{[t-1]_k} = \textsc{Svd}_k(N_{[t-1]}) = U_{(t-1)_k}\Sigma_{U_{(t-1)_k}} V_{U_{(t-1)_k}}^\top.$$

First observation is that $U_{(t-1)_k}$ is a "good" rank-k matrix to linearly represent all the points in $N_{[t-1]}$.[4] This follows from the observation that by setting $X = \Sigma_{U_{(t-1)_k}} V_{U_{(t-1)_k}}^\top$:

$$\sum_{\mathbf{y}_j\in N_{[t-1]}} \min_{\mathbf{x}_j} \|\mathbf{y}_j - U_{(t-1)_k}\mathbf{x}_j\|^2 = \min_X \|N_{[t-1]} - U_{(t-1)_k}X\|_F^2$$
$$\leq \|N_{[t-1]} - N_{[t-1]_k}\|_F^2.$$

The bound on $\|N_{[t-1]} - N_{[t-1]_k}\|_F^2$ follows from Theorem 1. In many practical scenarios, most of the mass from $N_{[t-1]}$ would be in its top k singular values (components), resulting in $\|N_{[t-1]} - N_{[t-1]_k}\|_F$ being small.

We can now use $U_{(t-1)_k}$ to detect anomalies in Y_t by following a simple approach. Since $U_{(t-1)_k}$ is a good basis to linearly reconstruct all the observed non-anomalous points in $Y_{[t-1]}$, we can use it to test whether a point $\mathbf{y}_i \in Y_t$ is "close" to space of non-anomalous points or not. This can be easily achieved by solving the following simple least-squares problem:

$$\min_{\mathbf{x}} \|\mathbf{y}_i - U_{(t-1)_k}\mathbf{x}\|. \tag{1}$$

As the columns of $U_{(t-1)_k}$ are orthogonal to each other, this least-squares problem has a simple closed-form solution

$$\mathbf{x}^* = (U_{(t-1)_k}^\top U_{(t-1)_k})^{-1} U_{(t-1)_k}^\top \mathbf{y}_i = U_{(t-1)_k}^\top \mathbf{y}_i.$$

The objective value of (1) at $\mathbf{x}^*$ is used as the anomaly score to decide if $\mathbf{y}_i$ is anomalous or not, with larger objective value denoting anomalies. In other words, the anomaly score for $\mathbf{y}_i$ is $\|(\mathbb{I}_m - U_{(t-1)_k}U_{(t-1)_k}^\top)\mathbf{y}_i\|$. Note that this anomaly score is exactly the length of the orthogonal projection of $\mathbf{y}_i$ onto the orthogonal complement $U_{(t-1)_k}$. This idea of using the projection of the data onto a residual subspace as means for detecting anomalies is quite popular and is also known to empirically work well [25, 23, 22].[5]

At time $t = 1$, to bootstrap the above anomaly detection framework, we gather a small training set of non-anomalous data (which is typically much easier to obtain than the anomalous data) and construct N_1 from it. Alternatively, if this is not possible, then we could collect a small set of data, apply any good unsupervised anomaly detection scheme to label the data, and then construct N_1 through it. Since this is just a onetime step, it

[2] In many social media, industrial applications, etc., many datapoints are generated simultaneously, therefore $n_t \gg 1$. Also not restricting n_t adds to the flexibility of our proposed framework.

[3] We defer the discussion on setting of k to later. Readers could think of k as a small number, much smaller than m.

[4] It is possible to use other (non-SVD) matrix factorization approaches to construct a basis matrix that can linearly represent $N_{[t-1]}$, however, using a low-rank SVD is attractive because it naturally comes with guarantees of Theorem 1.

[5] The prior works typically use residual of the principal component representation. Note that if the datapoints in $Y_{[t-1]}$ are centered, then $U_{(t-1)_k}$ directly relates to the top-k principal components.

Algorithm 1: AnomDetect (prototype algorithm for detecting anomalies at time t)

```
Input: $Y_t \in \mathbb{R}^{m \times n_t}$ (new observations),
       $U_{(t-1)_k} \in \mathbb{R}^{m \times k}$ (matrix with orthogonal columns),
       $\zeta \in \mathbb{R}$ (threshold parameter)
Anomaly score construction step:
$N_t \leftarrow [\,]$ , $\bar{N}_t \leftarrow [\,]$
for each point (column) $\mathbf{y}_i \in Y_t$ do
    Solve the following least-squares problem:
    $\mathbf{x}_i^* \leftarrow \operatorname{argmin}_{\mathbf{x}} \|\mathbf{y}_i - U_{(t-1)_k}\mathbf{x}\|$   $(\Longrightarrow \mathbf{x}_i^* = U_{(t-1)_k}^\top \mathbf{y}_i)$
    Define anomaly score: $a_i \leftarrow \|\mathbf{y}_i - U_{(t-1)_k}\mathbf{x}_i^*\|$
                          $(\Longrightarrow a_i = \|(\mathbb{I}_m - U_{(t-1)_k}U_{(t-1)_k}^\top)\mathbf{y}_i\|)$
    if $a_i \leq \zeta$ then
        $N_t \leftarrow [N_t, \mathbf{y}_i]$
    end
    else
        $\bar{N}_t \leftarrow [\bar{N}_t, \mathbf{y}_i]$ $(\Longrightarrow \mathbf{y}_i$ is marked as anomaly$)$
    end
end
$N_{[t]} \leftarrow [N_{[t-1]}, N_t]$
Updating the singular vectors:
Construct $U_{t_k} \in \mathbb{R}^{m \times k}$ such that:
  • it has orthogonal columns and
  • it equals/approximates the top-$k$ left singular vectors of $N_{[t]}$.
Algorithms 2, 3, or 4 could be used for this purpose
Return: $Y_t^{(g)}$, $\bar{N}_t$, and $U_{t_k}$
```

does not heavily influence the overall scalability of our proposed streaming approach.

Prototype Algorithm. In Algorithm AnomDetect, we present a simple prototype procedure for anomaly detection based on maintaining the left singular vectors (corresponding to the top-k singular values) of the streaming data. Since we have normalized all input points ($\mathbf{y}_i$'s) to have unit L_2-length, the objective values in (1) for all points are in the same scale. The Algorithm AnomDetect alternates between an anomaly detection and singular vector updating step. In the anomaly detection step, we use the past basis matrix to detect anomalies among the new incoming points by thresholding on the objective value of the least-squares problem (1). There are various ways for constructing the thresholds, which we will briefly discuss in Section 5. N_t is set of non-anomalous points in Y_t identified by the algorithm at time t.

We note here that our above framework is reminiscent to that used in *dictionary learning* where the goal is to estimate a collection of basis vectors over which a given data collection can be accurately reconstructed [32, 24]. In that context, $U_{(t-1)_k}$ is referred to as the dictionary matrix.

The main challenge is in updating the singular vectors. To start off, we first present an inefficient (baseline) approach based on global SVD updates, and later show how ideas from matrix sketching and randomized low-rank matrix approximations could be used to speedup the updating without any significant loss in quality of anomaly detection.

4.2 Global Algorithm (Baseline)

The simplest way of correctly updating the singular vectors is to simply (re)generate them from the globally collected sample set $N_{[t]} = [N_{[t-1]}, N_t]$. A more mature approach for incrementally and correctly generating the singular vectors of a matrix (with addition of new columns) based on the following lemma is outlined in Algorithm GlobalUpdate.

Lemma 3 ([6]). *Let $R = [P, Q]$, $\textsc{Svd}(R) = U_R \Sigma_R V_R^\top$, and $\textsc{Svd}(P) = U_P \Sigma_P V_P^\top$. Then $U_R = U_P U_F$, $\Sigma_R = \Sigma_F$, and where $F = [\Sigma_P, U_P^\top Q]$ and $\textsc{Svd}(F) = U_F \Sigma_F V_F^\top$.*

Algorithm 2: GlobalUpdate (global update of the left singular vectors at time t)

```
Input: $\hat{U}_{t-1}$, $\hat{\Sigma}_{t-1}$, $k$, and $N_t \in \mathbb{R}^{m \times n_t}$
$F \leftarrow [\hat{\Sigma}_{t-1}, \hat{U}_{t-1}^\top N_t]$
$U_F \Sigma_F V_F^\top \leftarrow \textsc{Svd}(F)$
$\hat{U}_t \leftarrow \hat{U}_{t-1} U_F$
$\hat{\Sigma}_t \leftarrow \Sigma_F$
$\hat{U}_{t_k} \leftarrow [\mathbf{u}_1, \ldots \mathbf{u}_k]$ (where $\hat{U}_t = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$)
Return: $\hat{U}_t$, $\hat{\Sigma}_t$, and $\hat{U}_{t_k}$
```

At time t, Algorithm GlobalUpdate takes $O(mn_{[t]})$ space and $O(\min\{m^2 n_{[t]}, mn_{[t]}^2\})$ time, where $n_{[t]}$ denotes the number of columns (datapoints) in the matrix $N_{[t]}$. It is obvious that a significant disadvantage of Algorithm GlobalUpdate is that both its computational and memory requirement increases with time. We overcome this problem by using matrix sketching. Our goal will be to show that while gaining in computational efficiency, the sketching approach still produces a good approximation to the top-k left singular vectors of $N_{[t]}$ at every time t.

4.3 Sketching-based Algorithms

In this section, we present two unsupervised streaming anomaly detection algorithms based on deterministic and randomized matrix sketching. Our algorithms build upon and improve Frequent Directions, a deterministic matrix sketching algorithm of Liberty [28]. The Frequent Directions algorithm operates in a streaming setting, and could be viewed as an extension of Misra and Gries approach for approximating frequency counts in a stream [34]. For completeness, we present the Frequent Directions algorithm in Appendix A (Algorithm 5). The inputs to the algorithm are an input data matrix $Z \in \mathbb{R}^{m \times n}$ and a sketch matrix $S \in \mathbb{R}^{m \times \ell}$ (which could be set to all zeros initially).[6] In each iteration, one column of Z is processed by the algorithm and the algorithm iteratively updates the matrix S such that for any unit vector $\mathbf{x} \in \mathbb{R}^m$, $\|Z^\top \mathbf{x}\|^2 - \|S^\top \mathbf{x}\|^2 \leq 2\|Z\|_F^2/\ell$. In other words, the sketched matrix has the guarantee that for any direction it is "close" to the input matrix. Practically too, the Frequent Directions algorithm seems to greatly outperform other commonly used sketching algorithms based on projection, hashing, and column sampling techniques [28, 14, 13, 12]. The computation time for this algorithm is dominated by a SVD computation in each iteration, which gives it a total running time of $O(mn\ell^2)$ (assuming $m \geq \ell$).

Recently, Ghashami *et al.* [14, 13], reanalyzed the Frequent Directions algorithm, to show that it provides relative error bounds for low-rank matrix approximation. Instead of S, their algorithm returns S_k (the rank-k approximation of S) and their main result shows that $\|Z_k\|_F^2 - \|S_k\|_F^2 \leq k/(\ell - k) \cdot \|Z - Z_k\|_F^2$, where Z_k is the rank-k approximation of Z.

In contrast to [28, 14, 13], where the sketch is updated after addition of every new column, we desire the sketch to be updated after addition of $n_t \geq 1$ columns. In our problem setup, at timestep t with $n_t \geq 1$ new columns, using Algorithm 5 for sketching would take $O(mn_t\ell^2)$ time. However, we show that an elegant trick of adding all the n_t columns simultaneously (instead

[6] The parameter $\ell > k$, but is generally much smaller than m.

of one at a time) and performing a low-rank SVD reduces the running time to $O(\max\{mn_t\ell, m\ell^2\})$ (again assuming $m \geq \ell$), without any loss in the sketching performance. Note that this resultant running time is $O(\min\{n_t, \ell\})$ times better than the running time ($O(mn_t\ell^2)$) of Frequent Directions. These ideas form the basis of our first sketching procedure described in Algorithm DetUpdate. The overall space complexity of Algorithm DetUpdate is $O(m \cdot \max_t\{n_t\} + m\ell)$, therefore the additional space overhead for the algorithm is only $O(m\ell)$ (as $O(m \cdot \max_t\{n_t\})$ space is needed to just read and store the input matrices).

Algorithm 3: DetUpdate (deterministic streaming update of the left singular vectors at time t)

```
    Input: $N_t \in \mathbb{R}^{m \times n_t}$, $k \leq \ell$, and $B_{t-1} \in \mathbb{R}^{m \times \ell}$ (the matrix
           sketch computed at time $t-1$)
1   $D_t \leftarrow [B_{t-1}, N_t]$
2   $\tilde{U}_{t_\ell}\tilde{\Sigma}_{t_\ell}\tilde{V}_{t_\ell}^\top \leftarrow \text{Svd}_\ell(D_t)$   (where $\tilde{\Sigma}_{t_\ell} = \text{diag}(\tilde{\sigma}_{t_1}, \dots, \tilde{\sigma}_{t_\ell})$)
3   $\tilde{\Sigma}_{t_\ell}^{(\text{trunc})} \leftarrow \text{diag}\left(\sqrt{\tilde{\sigma}_{t_1}^2 - \tilde{\sigma}_{t_\ell}^2}, \dots, \sqrt{\tilde{\sigma}_{t_{\ell-1}}^2 - \tilde{\sigma}_{t_\ell}^2}, 0\right)$
4   $B_t \leftarrow \tilde{U}_{t_\ell}\tilde{\Sigma}_{t_\ell}^{(\text{trunc})}$
5   $\tilde{U}_{t_k} \leftarrow [\mathbf{u}_1, \dots \mathbf{u}_k]$ (where $\tilde{U}_{t_\ell} = [\mathbf{u}_1, \dots, \mathbf{u}_\ell]$)
    Return: $B_t$ and $\tilde{U}_{t_k}$
```

In Algorithm DetUpdate, the matrix B_t is a sketch of the matrix $N_{[t]} = [N_1, \dots, N_t]$. Let B_{t_k} be the rank-k approximation of B_t. In the next section, we establish:

$$\|N_{[t]_k}\|_F^2 - \|B_{t_k}\|_F^2 \leq k/(\ell - k) \cdot \|N_{[t]} - N_{[t]_k}\|_F^2.$$

This proves that, while being computationally more efficient, the sketch matrices generated by Algorithm DetUpdate have the same guarantees as that generated by Frequent Directions [14].

Our second algorithm for matrix sketching is randomized, and stems from the observation that the low-rank SVD (Step 2) in Algorithm DetUpdate can be replaced by a randomized low-rank matrix approximation. This leads to even greater computational savings, however, as we note in the next section this efficiency gain comes at a cost of slightly higher error in sketching as compared to Algorithm DetUpdate (and Frequent Directions). Randomized low-rank matrix approximation has been a subject of lot of recent research with approaches based on sparsification, column selection, dimensionality reduction, etc., been devised for solving many matrix problems (see [16] and references therein). Here we adapt a technique suggested by Halko *et al.* [16] that is based on combining a randomized pre-processing step (multiplying by a random matrix and QR decomposition) along with a simple post-processing step (eigenvalue decomposition of a small matrix). The complete sketching procedure is described in Algorithm RandUpdate.

At timestep t, Algorithm RandUpdate takes $O(\ell T_{\text{mult}} + (m + n_t)\ell^2)$ time (assuming $m \geq \ell$), where T_{mult} denotes the cost of a matrix-vector multiplication with the input matrix M_t. The matrix-vector multiplication is a well-studied topic with numerous known efficient sequential/parallel algorithms. Note that this running time is smaller than that of Algorithm DetUpdate. which at timestep t takes $O(\max\{mn_t\ell, m\ell^2\})$ time. The overall space complexity of Algorithm RandUpdate is $O(m \cdot \max_t\{n_t\} + mr)$, therefore the additional space overhead for the algorithm is again only $O(mr) = O(m\ell)$.

Dealing with Concept Drift. An important feature of many real-world data streams is "concept drift", which means that the characteristics of the data can change over time. Algorithms for handling concept drift need to employ regular model updates as new data arrives. We refer the reader to the survey of [37] and the references therein for further motivation and background information on concept drift.

Algorithm 4: RandUpdate (randomized streaming update of the left singular vectors at time t)

```
Input: $N_t \in \mathbb{R}^{m \times n_t}$, $k \leq \ell$, and $E_{t-1} \in \mathbb{R}^{m \times \ell}$ (the
       randomized matrix sketch computed at time $t-1$)
$M_t \leftarrow [E_{t-1}, N_t]$
$r \leftarrow 100\ell$
Generate an $m \times r$ random Gaussian matrix $\Omega$
$Y \leftarrow M_t M_t^\top \Omega$
$QR \leftarrow \text{Qr}(Y)$ (QR factorization for $Y$)
$A_t\breve{\Sigma}_t^2 A_t^\top \leftarrow \text{Eig}(Q^\top M_t M_t^\top Q)$
                                  (where $\breve{\Sigma}_t^2 = \text{diag}(\breve{\sigma}_{t_1}^2, \dots, \breve{\sigma}_{t_r}^2)$)
$\breve{U}_t \leftarrow QA_t$   ($QQ^\top M_t M_t^\top QQ^\top$ approximates $M_t M_t^\top$)
$\breve{U}_{t_\ell} \leftarrow [\mathbf{u}_1, \dots, \mathbf{u}_\ell]$   (where $\breve{U}_{t_r} = [\mathbf{u}_1, \dots, \mathbf{u}_r]$ and $\ell \leq r$)
$\breve{\Sigma}_{t_\ell}^{(\text{trunc})} \leftarrow \text{diag}\left(\sqrt{\breve{\sigma}_{t_1}^2 - \breve{\sigma}_{t_\ell}^2}, \dots, \sqrt{\breve{\sigma}_{t_{\ell-1}}^2 - \breve{\sigma}_{t_\ell}^2}, 0\right)$
$E_t \leftarrow \breve{U}_{t_\ell}\breve{\Sigma}_{t_\ell}^{(\text{trunc})}$
$\breve{U}_{t_k} \leftarrow [\mathbf{u}_1, \dots \mathbf{u}_k]$ (where $\breve{U}_{t_\ell} = [\mathbf{u}_1, \dots, \mathbf{u}_\ell]$)
Return: $E_t$ and $\breve{U}_{t_k}$
```

In our formulation above concept-drift is well captured, as the underlying model is updated at each time t with all the identified non-anomalous points till time $t-1$ (related experiments in Section 5.3). Note that some applications might require anomalies to be detected based on a sliding window of inputs. Our algorithm could be easily adapted to these scenarios by modifying the matrix sketch construction. A simple idea, in case of the sliding window of length w, is to at every time t maintain a separate sketch of the non-anomalous data identified over the time interval $[t - w + j, t - 1]$ (i.e., $N_{t-w+j}, \dots, N_{t-1}$) for each $j \in \{11, \dots, w - 1\}$, and update all these $(w - 1)$ different sketches with N_t. This ensures that at every time $t + 1$, we have the desired sketch from $[N_{t-w+1}, \dots, N_t]$. This only requires a small amount of additional bookkeeping, and this idea can be efficiently implemented. Due to space limitations, we defer further details to the full version of this paper.

4.4 Anomaly Detection Performance

Analysis of Algorithm RandUpdate. In this section, we theoretically compare the anomaly detection performance obtained by using the matrix $\breve{U}_{t_k}$ (output of Algorithm RandUpdate) in Algorithm AnomDetect to that of using the true singular vector matrix $\hat{U}_{t_k}$ (output of Algorithm GlobalUpdate). The analysis of Algorithm DetUpdate is a special subcase of this analysis, and we present that later.

Overall, the analysis can be split into two main parts: 1) we show that Algorithm RandUpdate produces good matrix sketches (Lemma 9) at every time t, and 2) we combine the guarantee on the sketches plus techniques from matrix perturbation theory to compare the anomaly detection performance of using $\breve{U}_{t_k}$ in Algorithm AnomDetect as compared to $\hat{U}_{t_k}$ (Theorem 14).

If $\breve{U}_{t_k}$ is used in Algorithm AnomDetect, then the anomaly score for any input point $\mathbf{y} \in \mathbb{R}^m$ arriving at time $t+1$ is

$$\|\mathbf{y} - \breve{U}_{t_k}\mathbf{x}_s^*\|, \text{ where } \mathbf{x}_s^* = \text{argmin}_{\mathbf{x}}\|\mathbf{y} - \breve{U}_{t_k}\mathbf{x}\|.$$

Similarly if $\hat{U}_{t_k}$ is used in Algorithm AnomDetect, then the anomaly score for $\mathbf{y}$ is

$$\|\mathbf{y} - \hat{U}_{t_k}\mathbf{x}_g^*\|, \text{ where } \mathbf{x}_g^* = \text{argmin}_{\mathbf{x}}\|\mathbf{y} - \hat{U}_{t_k}\mathbf{x}\|.$$

We start with a simple observation (proof omitted here), that a bound on $\|\hat{U}_{t_k} - \breve{U}_{t_k}\|_F$ directly translates into a bound on the difference between these two anomaly scores.

LEMMA 4. *Let* $\mathbf{x}_g^* = \textit{argmin}_{\mathbf{x}}\|\mathbf{y} - \hat{U}_{t_k}\mathbf{x}\|$, $\mathbf{x}_s^* = \textit{argmin}_{\mathbf{x}}\|\mathbf{y} - \breve{U}_{t_k}\mathbf{x}\|$. *Then,*

$$\Big|\|\mathbf{y} - \hat{U}_{t_k}\mathbf{x}_g^*\| - \|\mathbf{y} - \breve{U}_{t_k}\mathbf{x}_s^*\|\Big| \leq \|\hat{U}_{t_k} - \breve{U}_{t_k}\|_F.$$

We now concentrate on bounding $\|\hat{U}_{t_k} - \breve{U}_{t_k}\|_F$. To do so, we first construct a bound on $\|N_{[t]_k}N_{[t]_k}^\top - E_{t_k}E_{t_k}^\top\|_F$. We then plug this bound into a recent matrix perturbation result to show that $\hat{U}_{t_k}$ (the eigenvectors of $N_{[t]_k}N_{[t]_k}^\top$) and $\breve{U}_{t_k}$ (the eigenvectors of $E_{t_k}E_{t_k}^\top$) are close. Our analysis relies on the following result from Halko *et al.* [16] that bounds the error due to the randomized low-rank approximation.

THEOREM 5. *(Restated from Corollary 10.9 of [16]) Consider Algorithm* RANDUPDATE *at time t. Let* $\operatorname{diag}(\bar{\sigma}_{t_1}, \ldots, \bar{\sigma}_{t_m})$ *be the eigenvalues of* $M_tM_t^\top$, *then with probability at least* $1 - 6e^{-99\ell}$, $\|M_tM_t^\top - \breve{U}_t\breve{\Sigma}_t^2\breve{U}_t\| \leq 38\bar{\sigma}_{t_{\ell+1}} + 2(\sum_{i=\ell+1}^m \bar{\sigma}_{t_i}^2)^{1/2}/\sqrt{\ell}$.

We will need a few additional notations:

(a) $E_{t_k} = \breve{U}_{t_k}\breve{\Sigma}_{t_k}^{(\mathrm{trunc})}$ (rank-k approximation of E_t),

(b) $\breve{\Delta}_t = \sum_{j=1}^t \breve{\sigma}_{j_\ell}^2$,

(c) $\upsilon_j = 38\bar{\sigma}_{j_{\ell+1}} + 2\left(\sum_{i=\ell+1}^m \bar{\sigma}_{j_i}^2\right)^{1/2}/\sqrt{\ell}$ (error bound from Theorem 5, at time j),

(d) $\Upsilon_t = \sum_{j=1}^t \upsilon_j$

(e) $\kappa = \sigma_1(N_{[t]})/\sigma_k(N_{[t]})$, where $\sigma_1(N_{[t]}) \geq \cdots \geq \sigma_m(N_{[t]})$ are the singular values of $N_{[t]}$,

(f) $R_t = QQ^\top M_t$,

(g) $P_t = QA_t\breve{\Sigma}_t = \breve{U}_t\breve{\Sigma}_t$ (by construction in Algorithm RANDUPDATE, $R_tR_t^\top = P_tP_t^\top$).

As columns of Q are orthogonal to each other, $QQ^\top$ is a projection matrix, and therefore by standard properties of projection matrices and noting that $(QQ^\top)^\top = QQ^\top$,

$$\|M_t\|_F^2 \geq \|QQ^\top M_t\|_F^2 = \|R_t\|_F^2 = \|P_t\|_F^2. \qquad (2)$$

Similarly for all unit vectors $\mathbf{x} \in \mathbb{R}^m$,

$$\|M_t^\top\mathbf{x}\|^2 \geq \|(QQ^\top M_t)^\top\mathbf{x}\|^2 = \|R_t^\top\mathbf{x}\|^2 = \|P_t^\top\mathbf{x}\|^2. \qquad (3)$$

For ease of presentation, in the following, we are going to assume, that $t\cdot 6e^{-99\ell} \ll 1$.[7] Note that $e^{-99\ell}$ is a very tiny number.

LEMMA 6. *At time t, Algorithm* RANDUPDATE *maintains that:* $\|N_{[t]}\|_F^2 - \|E_t\|_F^2 \geq \ell\breve{\Delta}_t$.

PROOF. At time t, $\|M_t\|_F^2 = \|E_{t-1}\|_F^2 + \|N_t\|_F^2$. We also have $\|P_t\|_F^2 \geq \|E_t\|_F^2 + \ell\breve{\sigma}_{t_\ell}^2$. Since, $\|M_t\|_F^2 \geq \|P_t\|_F^2$ (from (2)), we have $\|M_t\|_F^2 \geq \|E_t\|_F^2 + \ell\breve{\sigma}_{t_\ell}^2$. Solving for $\|N_t\|_F^2$ from these inequalities and summing over $j \leq t$, we get,

$$\|N_{[t]}\|_F^2 = \sum_{j=1}^t \|N_j\|_F^2$$
$$\geq \sum_{j=1}^t (\|E_j\|_F^2 - \|E_{j-1}\|_F^2 + \ell\breve{\sigma}_{j_\ell}^2) \geq \|E_t\|_F^2 + \ell\breve{\Delta}_t.$$

The last line follows as E_0 is all zeros matrix. □

The following lemma shows that for any direction $\mathbf{x}$, $N_{[t]}$ and E_t are with high probability not too far apart.

LEMMA 7. *For any unit vector* $\mathbf{x} \in \mathbb{R}^m$, *at any time t,* $0 \leq \|N_{[t]}^\top\mathbf{x}\|^2 - \|E_t^\top\mathbf{x}\|^2$, *and with probability at least* $1 - t\cdot 6e^{-99\ell}$,

$$\|N_{[t]}^\top\mathbf{x}\|^2 - \|E_t^\top\mathbf{x}\|^2 \leq \breve{\Delta}_t + \Upsilon_t.$$

PROOF. To show $|N_{[t]}^\top\mathbf{x}\|^2 - \|E_t^\top\mathbf{x}\|^2 > 0$, observe that

$$\|E_{t-1}^\top\mathbf{x}\|^2 + \|N_t^\top\mathbf{x}\|^2 = \|M_t^\top\mathbf{x}\|^2.$$

Since $\|P_t^\top\mathbf{x}\|^2 \geq \|E_t^\top\mathbf{x}\|^2$ (by construction) and $\|M_t^\top\mathbf{x}\|^2 \geq \|P_t^\top\mathbf{x}\|^2$ (from (3)), we have,

$$\|N_{[t]}^\top\mathbf{x}\|^2 = \sum_{j=1}^t \|N_j^\top\mathbf{x}\|^2 \geq \sum_{j=1}^t (\|E_j^\top\mathbf{x}\|^2 - \|E_{j-1}^\top\mathbf{x}\|^2) \geq \|E_t^\top\mathbf{x}\|^2.$$

Here we used that E_0 is an all zeros matrix. Now let us concentrate on showing

$$\|N_{[t]}^\top\mathbf{x}\|^2 - \|E_t^\top\mathbf{x}\|^2 \leq \Upsilon_t + \breve{\Delta}_t.$$

Let $\mathbf{u}_i$ be the ith column in $\breve{U}_t$. $\breve{\sigma}_{t_i}^2 - \breve{\sigma}_{t_\ell}^2$ is the ith singular value of E_t. Let $R_p = \operatorname{rank}(P_t)$.

$$\|P_t^\top\mathbf{x}\|^2 = \sum_{i=1}^{R_p} \breve{\sigma}_{t_i}^2\langle\mathbf{u}_i, \mathbf{x}\rangle^2 = \sum_{i=1}^{R_p}(\breve{\sigma}_{t_i}^2 + \breve{\sigma}_{t_\ell}^2 - \breve{\sigma}_{t_\ell}^2)\langle\mathbf{u}_i, \mathbf{x}\rangle^2$$
$$= \sum_{i=1}^{R_p}(\breve{\sigma}_{t_i}^2 - \breve{\sigma}_{t_\ell}^2)\langle\mathbf{u}_i, \mathbf{x}\rangle^2 + \sum_{i=1}^{R_p}\breve{\sigma}_{t_\ell}^2\langle\mathbf{u}_i, \mathbf{x}\rangle^2$$
$$\leq \sum_{i=1}^{\ell}(\breve{\sigma}_{t_i}^2 - \breve{\sigma}_{t_\ell}^2)\langle\mathbf{u}_i, \mathbf{x}\rangle^2 + \breve{\sigma}_{t_\ell}^2\sum_{i=1}^{R_p}\langle\mathbf{u}_i, \mathbf{x}\rangle^2 \leq \|E_t^\top\mathbf{x}\|^2 + \breve{\sigma}_{t_\ell}^2.$$

For the first inequality we used that for $i > \ell$, $\breve{\sigma}_{t_i}^2 \leq \breve{\sigma}_{t_\ell}^2$. For the second inequality, we use that $\sum_{i=1}^{R_p}\langle\mathbf{u}_i, \mathbf{x}\rangle^2 \leq \|\mathbf{x}\|^2 = 1$ (as $\mathbf{x}$ is a unit vector).

Since for all unit vectors $\mathbf{x} \in \mathbb{R}^m$, $\|M_t^\top\mathbf{x}\|^2 - \|P_t^\top\mathbf{x}\|^2 \leq \|M_tM_t^\top - P_tP_t^\top\|$, we get with probability at least $1 - 6e^{-99\ell}$,

$$\|M_t^\top\mathbf{x}\|^2 \leq \|P_t^\top\mathbf{x}\|^2 + \|M_tM_t^\top - P_tP_t^\top\| = \|P_t^\top\mathbf{x}\|^2 + \upsilon_t.$$

Since $\|M_t^\top\mathbf{x}\|^2 = \|E_{t-1}^\top\mathbf{x}\|^2 + \|N_t^\top\mathbf{x}\|^2$, we get with probability at least $1 - 6e^{-99\ell}$,

$$\|E_{t-1}^\top\mathbf{x}\|^2 + \|N_t^\top\mathbf{x}\|^2 \leq \upsilon_t + \|E_t^\top\mathbf{x}\|^2 + \breve{\sigma}_{t_\ell}^2.$$

Subtracting $\|E_{t-1}^\top\mathbf{x}\|^2$ from both sides and summing over $j \leq t$ with a union bound for probabilities, we get that with probability at least $1 - t\cdot 6e^{-99\ell}$,

$$\|N_{[t]}^\top\mathbf{x}\|^2 = \sum_{j=1}^t \|N_j^\top\mathbf{x}\|^2$$
$$\leq \sum_{j=1}^t (\|E_j^\top\mathbf{x}\|^2 - \|E_{j-1}^\top\mathbf{x}\|^2 + \breve{\sigma}_{j_\ell}^2 + \upsilon_j) = \|E_t^\top\mathbf{x}\|^2 + \breve{\Delta}_t + \Upsilon_t.$$

Again we used that E_0 is an all zeros matrix. □

Since for all unit vectors $\mathbf{x} \in \mathbb{R}^m$,

$$\|N_{[t]}^\top\mathbf{x}\|^2 - \|E_t^\top\mathbf{x}\|^2 \geq 0 \Longrightarrow N_{[t]}N_{[t]}^\top \succeq E_tE_t^\top.$$

From Claim 2, for all unit vectors $\mathbf{x} \in \mathbb{R}^m$, $\kappa\|N_{[t]_k}^\top\mathbf{x}\| \geq \|N_{[t]}^\top\mathbf{x}\|$. Therefore,

$$\kappa^2 N_{[t]_k}N_{[t]_k}^\top \succeq N_{[t]}N_{[t]}^\top \succeq E_tE_t^\top \succeq E_{t_k}E_{t_k}^\top.$$

LEMMA 8. *Let* $N_{[t]_k}$ *be the best rank-k approximation to* $N_{[t]}$. *Then with probability at least* $1 - t\cdot 6e^{-99\ell}$,

$$\breve{\Delta}_t \leq \frac{\|N_{[t]} - N_{[t]_k}\|_F^2 + k\Upsilon_t}{\ell - k}.$$

[7] If this inequality is violated, then one could use a slightly larger r in Step 2 of Algorithm RANDUPDATE.

Proof. From Lemma 6, $\|N_{[t]}\|_F^2 - \|E_t\|_F^2 \geq \ell\check{\Delta}_t$. Let $R_y = \text{rank}(N_{[t]})$ and $\mathbf{v}_1, \dots, \mathbf{v}_{R_y}$ be the left singular vectors of $N_{[t]}$ corresponding to the non-zero singular values of $N_{[t]}$, we have with probability at least $1 - t \cdot 6e^{-99\ell}$,

$$\begin{aligned}
\ell\check{\Delta}_t &\leq \|N_{[t]}\|_F^2 - \|E_t\|_F^2 \\
&= \sum_{i=1}^{k} \|N_{[t]}^\top \mathbf{v}_i\|^2 + \sum_{i=k+1}^{R_y} \|N_{[t]}^\top \mathbf{v}_i\|^2 - \|E_t\|_F^2 \\
&= \sum_{i=1}^{k} \|N_{[t]}^\top \mathbf{v}_i\|^2 + \|N_{[t]} - N_{[t]_k}\|_F^2 - \|E_t\|_F^2 \\
&\leq \sum_{i=1}^{k} \|N_{[t]}^\top \mathbf{v}_i\|^2 + \|N_{[t]} - N_{[t]_k}\|_F^2 - \sum_{i=1}^{k} \|E_t^\top \mathbf{v}_i\|^2 \\
&= \|N_{[t]} - N_{[t]_k}\|_F^2 + \sum_{i=1}^{k} (\|N_{[t]}^\top \mathbf{v}_i\|^2 - \|E_t^\top \mathbf{v}_i\|^2) \\
&\leq \|N_{[t]} - N_{[t]_k}\|_F^2 + k(\Upsilon_t + \check{\Delta}_t).
\end{aligned}$$

First inequality uses that $\sum_{i=1}^{k} \|E_t^\top \mathbf{v}_i\|^2 \leq \|E_t\|_F^2$, and the last inequality is based on Lemma 7. Solving for $\check{\Delta}_t$ in the above inequality gives the claimed result. □

Using Lemma 8, we can relate $\|N_{[t]_k}\|_F^2$ to $\|E_{t_k}\|\|_F^2$ to show that E_{t_k} is a "good" sketch of $N_{[t]_k}$.

Lemma 9. *At any time t, $0 \leq \|N_{[t]_k}\|_F^2 - \|E_{t_k}\|_F^2$, and with probability at least $1 - t \cdot 6e^{-99\ell}$,*

$$\|N_{[t]_k}\|_F^2 - \|E_{t_k}\|_F^2 \leq k\Upsilon_t + \frac{k(\|N_{[t]} - N_{[t]_k}\|_F^2 + k\Upsilon_t)}{\ell - k}.$$

Proof. Let $\mathbf{v}_1, \dots, \mathbf{v}_k$ and $\mathbf{u}_1, \dots, \mathbf{u}_k$ be the left singular vectors of $N_{[t]}$ and E_t corresponding to their top-k singular values. We have

$$\begin{aligned}
\|N_{[t]_k}\|_F^2 = \sum_{i=1}^{k} \|N_{[t]}^\top \mathbf{v}_i\|^2 &\geq \sum_{i=1}^{k} \|N_{[t]}^\top \mathbf{u}_i\|^2 \\
&\geq \sum_{i=1}^{k} \|E_t^\top \mathbf{u}_i\|^2 = \|E_{t_k}\|_F^2.
\end{aligned}$$

This proves that $0 \leq \|N_{[t]_k}\|_F^2 - \|E_{t_k}\|_F^2$. The upper bound can be established by noticing that with probability at least $1 - t \cdot 6e^{-99\ell}$,

$$\begin{aligned}
\|E_{t_k}\|_F^2 \geq \sum_{i=1}^{k} \|E_{t_k}^\top \mathbf{v}_i\|^2 &\geq \sum_{i=1}^{k} (\|N_{[t]}^\top \mathbf{v}_i\|^2 - \Upsilon_t - \check{\Delta}_t) \\
&= \|N_{[t]_k}\|_F^2 - k\Upsilon_t - k\check{\Delta}_t,
\end{aligned}$$

where the second inequality follows from Lemma 7. Now substituting for $\check{\Delta}_t$ from Lemma 8 gives the result. □

Using this above lemma and the fact that $\kappa^2 N_{[t]_k} N_{[t]_k}^\top \succeq E_{t_k} E_{t_k}^\top$, we can prove the following proposition.

Proposition 10. *At time t, E_{t_k} (rank-k approximation of E_t generated by Algorithm* RandUpdate*) satisfies,*

$$\|\kappa^2 N_{[t]_k} N_{[t]_k}^\top - E_{t_k} E_{t_k}^\top\|_F \leq \kappa^2 \|N_{[t]_k}\|_F^2 - \|E_{t_k}\|_F^2.$$

Proof. For a positive semidefinite matrix, the trace is greater than or equal to the Frobenius norm. Since, we have established that $\kappa^2 N_{[t]_k} N_{[t]_k}^\top - E_{t_k} E_{t_k}^\top$ is a positive semidefinite matrix.

$$\begin{aligned}
&\|\kappa^2 N_{[t]_k} N_{[t]_k}^\top - E_{t_k} E_{t_k}^\top\|_F \leq \text{tr}(\kappa^2 N_{[t]_k} N_{[t]}^\top - E_{t_k} E_{t_k}^\top) \\
&= \kappa^2 \text{tr}\left(N_{[t]_k} N_{[t]_k}^\top\right) - \text{tr}(E_{t_k} E_{t_k}^\top) = \kappa^2 \|N_{[t]_k}\|_F^2 - \|E_{t_k}\|_F^2.
\end{aligned}$$

The first inequality follows from the trace-Frobenius inequality of positive semidefinite matrices. □

We need couple of more definitions. Define Φ_a as,

$$\Phi_a = \frac{\kappa^2 \|N_{[t]_k}\|_F^2 - \|E_{t_k}\|_F^2}{\|N_{[t]_k}\|_F^2 - \|E_{t_k}\|_F^2}. \tag{4}$$

Note that $\Phi_a \geq 1$ as $\|N_{[t]_k}\|_F^2 \geq \|E_{t_k}\|_F^2$ (from Lemma 9). In fact, for small k's (as in our setting), typically κ (the ratio between the largest and kth largest singular value of $N_{[t]}$) is bounded, yielding $\Phi_a = O(1)$.

Define Φ_b as,

$$\Phi_b = \frac{\|\kappa^2 N_{[t]} N_{[t]}^\top - E_t E_t^\top\|}{\|\kappa^2 N_{[t]_k} N_{[t]_k}^\top - E_{t_k} E_{t_k}^\top\|}. \tag{5}$$

Claim 11.

$$\Phi_b \leq 1 + 2/(\kappa^2 - \|E_t\|^2/\|N_{[t]}\|^2).$$

Proof. A straightforward manipulation shows that the numerator of Φ_b,

$$\begin{aligned}
\|\kappa^2 N_{[t]} N_{[t]}^\top - E_t E_t^\top\| &\leq \kappa^2 \|N_{[t]} N_{[t]}^\top - N_{[t]_k} N_{[t]_k}^\top\| \\
&+ \|E_t E_t^\top - E_{t_k} E_{t_k}^\top\| + \|\kappa^2 N_{[t]_k} N_{[t]_k}^\top - E_{t_k} E_{t_k}^\top\|.
\end{aligned} \tag{6}$$

Note that using Theorem 1,

$$\|N_{[t]} N_{[t]}^\top - N_{[t]_k} N_{[t]_k}^\top\| = \sigma_{k+1}^2,$$

where σ_{k+1} is the $(k+1)$st singular value of $N_{[t]}$. Similarly by using Theorem 1, $\|E_t E_t^\top - E_{t_k} E_{t_k}^\top\|$ is equal to the square of the $(k+1)$st singular value of E_t. Since we have already established $N_{[t]} N_{[t]}^\top - E_t E_t^\top \succeq 0$, this implies that $\|E_t E_t^\top - E_{t_k} E_{t_k}^\top\| \leq \sigma_{k+1}^2$. Let $\|N_{[t]}\| = \sigma_1$. Substituting these observations into Φ_b:

$$\Phi_b \leq 1 + \frac{(\kappa^2 + 1)\sigma_{k+1}^2}{\|\kappa^2 N_{[t]_k} N_{[t]_k}^\top - E_{t_k} E_{t_k}^\top\|} \leq 1 + \frac{(\kappa^2 + 1)\sigma_{k+1}^2}{\kappa^2 \sigma_1^2 - \|E_t\|^2}.$$

The last inequality follows as by Weyl's inequality [15] the largest eigenvalue of $\|\kappa^2 N_{[t]_k} N_{[t]_k}^\top - E_{t_k} E_{t_k}^\top\|$ is greater than equal to $\kappa^2 \|N_{[t]_k}\|^2 - \|E_{t_k}\|^2$. We also used that $\|N_{[t]_k}\|^2 = \|N_{[t]}\|^2$ and $\|E_t\|^2 = \|E_{t_k}\|^2$. Since, $\kappa \leq \sigma_1/\sigma_{k+1}$, bound on Φ_b can be re-expressed as,

$$\Phi_b \leq 1 + \frac{(\kappa^2 + 1)\frac{\sigma_1^2}{\kappa^2}}{\kappa^2 \sigma_1^2 - \|E_t\|^2} \leq 1 + \frac{2}{\kappa^2 - \|E_t\|^2/\sigma_1^2}.$$

Here we used that $(\kappa^2 + 1)/\kappa^2 \leq 2$ as $\kappa \geq 1$. □

Note that $\|E_t\|^2 \leq \|N_{[t]}\|^2$ (as $N_{[t]} N_{[t]}^\top \succeq E_t E_t^\top$) . Typically κ is also bounded away from 1, yielding $\Phi_b = O(1)$.

We now use the theory of matrix perturbation to relate $\breve{U}_{t_k}$ (from Algorithm RandUpdate) to $\hat{U}_{t_k}$ (true left singular vectors corresponding to top-k singular values of $N_{[t]}$). There is lot of prior work in matrix perturbation theory that relates the eigenvalues, singular values, eigenspaces, and singular subspaces, etc., of the matrix $Z + Z'$ to the corresponding quantity in Z, under various conditions on the matrices Z and Z'. Here we use a recent result from Chen *et al.* [10] that studies behavior of the eigenvector matrix of a Hermitian (symmetric) matrix under a small perturbation.

Theorem 12. *(Restated from Theorem 2.1 [10]) Let $A \in \mathbb{R}^{m \times m}$ be a symmetric matrix with distinct eigenvalues with* Eig$(A) = U\Lambda U^\top$ *where $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_m)$. Let $A_{\text{per}} = A + \Phi$ be a symmetric matrix. Let $L = L(\Lambda) = \min_{i \neq j} |\lambda_i - \lambda_j| > 0$, $\beta = \|\Phi\|_F/L$, and $\alpha = 2\|A\|/L$, with β satisfying: $\beta \leq 1/(1 + 4\alpha)$. Then* Eig$(A_{\text{per}}) = U_{\text{per}}\Lambda_{\text{per}}U_{\text{per}}^\top$ *such that $\|U - U_{\text{per}}\|_F \leq \sqrt{2}\beta/(1 + 4\alpha^2)^{1/4}$.*

We now can apply Proposition 10 and Theorem 12 to bound $\|\hat{U}_{t_k} - \breve{U}_{t_k}\|_F$. We do so by constructing matrices: $A = \kappa^2 N_{[t]} N_{[t]}^\top$ and $A_{\text{per}} = E_t E_t^\top$. Let ℓ be such that:

$$\frac{\sqrt{m}\Phi_b\Phi_a k\Upsilon_t}{L} + \frac{\sqrt{m}\Phi_b\Phi_a k(\|N_{[t]} - N_{[t]_k}\|_F^2 + k\Upsilon_t)}{(\ell - k)L} \leq \frac{L}{L + 4\kappa^2\|N_{[t]}\|^2}. \quad (7)$$

An important point to note in the above equation (7) is that both terms in the left-hand side are monotonically decreasing functions in ℓ (for the first term, Υ_t decreases with ℓ).

Lemma 13. *Let λ_i be the ith eigenvalue of $N_{[t]}N_{[t]}^\top$ and $L = \min_{i \neq j} |\lambda_i - \lambda_j| > 0$. If ℓ satisfies (7) for $\Upsilon_t, \Phi_a, \Phi_b$ defined in* **(4)**, (4), (5) *respectively, then with probability at least $1 - t \cdot 6e^{-99\ell}$,*

$$\|\hat{U}_{t_k} - \breve{U}_{t_k}\|_F \leq \frac{\sqrt{2}L}{(\sqrt{L + 8\kappa^2\|N_{[t]}\|^2}\sqrt[4]{L^2 + 16\kappa^4\|N_{[t]}\|^4})}.$$

Proof. Set $A = \kappa^2 N_{[t]}N_{[t]}^\top$ and $A_{\text{per}} = E_t E_t^\top$. Now $\alpha = 2\|A\|/L$. Concentrating on β, with probability at least $1 - t \cdot 6e^{-99\ell}$,

$$\begin{aligned}\beta = \frac{\|A - A_{\text{per}}\|_F}{L} &\leq \frac{\sqrt{m}\Phi_b\|\kappa^2 N_{[t]_k}N_{[t]_k}^\top - E_{t_k}E_{t_k}^\top\|_F}{L}\\ &\leq \frac{\sqrt{m}\Phi_b(\kappa^2\|N_{[t]_k}\|_F^2 - \|E_{t_k}\|_F^2)}{L}\\ &= \frac{\sqrt{m}\Phi_b\Phi_a(\|N_{[t]_k}\|_F^2 - \|E_{t_k}\|_F^2)}{L}\\ &\leq \frac{\sqrt{m}\Phi_b\Phi_a k\Upsilon_t}{L} + \frac{\sqrt{m}\Phi_b\Phi_a k(\|N_{[t]} - N_{[t]_k}\|_F^2 + k\Upsilon_t)}{(\ell - k)L}.\end{aligned}$$

The last inequality follows from Lemma 9. To apply Theorem 12, we need to satisfy the condition of $\beta \leq 1/(1 + 4\alpha)$. This translates to setting ℓ to satisfy (7) (assuming $k < \sqrt{m}$ and the Lemma 9 holds), The eigendecomposition of $N_{[t]}N_{[t]}^\top$ is: $N_{[t]}N_{[t]}^\top = \hat{U}_t\hat{\Sigma}_t\hat{U}_t^\top$. Similarly the eigendecomposition of $E_tE_t^\top$ is:

$$E_tE_t^\top = [\breve{U}_t|\mathbf{o}_{r+1}, \ldots, \mathbf{o}_m]\times$$
$$\text{diag}(\breve{\sigma}_{t_1}^2 - \breve{\sigma}_{t_\ell}^2, \ldots, \breve{\sigma}_{t_{\ell-1}}^2 - \breve{\sigma}_{t_\ell}^2, 0, \ldots, 0) \times [\breve{U}_t|\mathbf{o}_{r+1}, \ldots, \mathbf{o}_m]^\top,$$

where $[\breve{U}_t|\mathbf{o}_{r+1}, \ldots, \mathbf{o}_m]$ is an $m \times m$ orthogonal matrix. Note that $\breve{U}_t$ is an $m \times r$ matrix. The choice of $\mathbf{o}_{r+1}, \ldots, \mathbf{o}_m$ does not matter here.

Substituting the values of $\beta \leq 1/(1 + 4\alpha)$ and α, we have by the bound of Theorem 12,

$$\begin{aligned}&\|\hat{U}_t - [\breve{U}_t|\mathbf{o}_{r+1}, \ldots, \mathbf{o}_m]\|_F\\ &\leq \frac{\sqrt{2}L}{\sqrt{L + 8\kappa^2\|N_{[t]}\|^2}\sqrt[4]{L^2 + 16\kappa^4\|N_{[t]}\|^4}}.\end{aligned}$$

Noting that

$$\|\hat{U}_{t_k} - \breve{U}_{t_k}\|_F \leq \|\hat{U}_t - [\breve{U}_t|\mathbf{o}_{r+1}, \ldots, \mathbf{o}_m]\|_F$$

as $\hat{U}_{t_k} - \breve{U}_{t_k}$ is a submatrix of $\hat{U}_t - [\breve{U}_t|\mathbf{o}_{r+1}, \ldots, \mathbf{o}_m]$ (remember $k \leq \ell \leq r$) completes the proof. □

Neither the numerical constants nor the precise form of the bound on ℓ in (7) are optimal because of the slackness in Theorem 12. The bound on ℓ in (7) could be simplified a bit for some interesting cases, e.g., when k is small and $1 < \kappa \leq O(1)$ then $\Gamma_a = O(1)$ and $\Gamma_b = O(1)$. The assumption of $L > 0$ is something that is commonly satisfied in practice, especially if m is reasonably smaller than the number of datapoints in $N_{[t]}$.

We can now compare the anomaly scores generated by using either $\hat{U}_{t_k}$ or $\breve{U}_{t_k}$ in Algorithm AnomDetect. The theorem follows from Lemmas 4 and 13. Informally, the theorem shows that under some reasonable assumptions and settings of parameters, we can use the efficient Algorithm RandUpdate for singular value updating in Algorithm AnomDetect and still obtain anomaly scores that are close to that obtained using the true (actual) singular vectors. The theorem relies on the following two assumptions:

(A1) $L = \min_{i \neq j} |\lambda_i - \lambda_j| > 0$ where λ_i be the ith eigenvalue of $N_{[t]}N_{[t]}^\top$.

(A2) ℓ satisfies the bound from (7).

Theorem 14 (Main Theorem). *Let $N_1, \ldots, N_t$ be a sequence of matrices with $N_{[t]} = [N_1, \ldots, N_t]$. Let $N_{[t]_k} = \hat{U}_{t_k}\hat{\Sigma}_{t_k}\hat{V}_{t_k}^\top$ be the best rank-k approximation to $N_{[t]}$. Then for any unit vector $\mathbf{y} \in \mathbb{R}^m$, $\breve{U}_{t_k}$ (generated by the Algorithm* RandUpdate*), under assumptions* (A1) *and* (A2)*, with probability at least $1 - t \cdot 6e^{-99\ell}$, satisfies:*

$$\begin{aligned}&\left|\min_{\mathbf{x}\in\mathbb{R}^k}\|\mathbf{y} - \hat{U}_{t_k}\mathbf{x}\| - \min_{\mathbf{x}\in\mathbb{R}^k}\|\mathbf{y} - \breve{U}_{t_k}\mathbf{x}\|\right|\\ &\leq \frac{\sqrt{2}L}{(\sqrt{L + 8\kappa^2\|N_{[t]}\|^2}\sqrt[4]{L^2 + 16\kappa^4\|N_{[t]}\|^4})}.\end{aligned}$$

The above bound on the difference in anomaly scores is an increasing function in L.

Remark 15. *Note that in Theorem 14, we have assumed that the set of matrices $N_1, \ldots, N_t$ given to both algorithms are the same. This assumption is important for any theoretical comparison between the algorithms. The foundation for this assumption comes from the following inductive observation: by Theorem 14, at time $t + 1$, for each point in Y_{t+1}, the anomaly scores constructed by using either matrices $\hat{U}_{t_k}$ or $\tilde{U}_{t_k}$ are "almost" the same, therefore, N_{t+1} generated by using either $\hat{U}_{t_k}$ or $\tilde{U}_{t_k}$ in Algorithm* AnomDetect *are also almost the same.*

Analysis of Algorithm DetUpdate. The analysis is identical to that of Algorithm RandUpdate. Since the SVD in Step 2 of Algorithm DetUpdate is exact, the error due to randomization (Υ_t) is zero. Let B_{t_k} be the rank-k approximation of B_t. Define Γ_a and Γ_b by replacing E_t with B_t and E_{t_k} with B_{t_k} in the definitions of Φ_a (4) and Φ_b (5), respectively.

$$\Gamma_a = \frac{\kappa^2\|N_{[t]_k}\|_F^2 - \|B_{t_k}\|_F^2}{\|N_{[t]_k}\|_F^2 - \|B_{t_k}\|_F^2} \text{ and } \Gamma_b = 1 + \frac{2}{\kappa^2 - \|B_t\|^2/\|N_{[t]}\|^2}.$$

For Algorithm DetUpdate, the requirement on ℓ needed for the application of the matrix perturbation bound of Theorem 12 simplifies to:

$$\ell = \Omega\left(\frac{\sqrt{m}\kappa^2\|N_{[t]}\|^2\Gamma_a\Gamma_b k\|N_{[t]} - N_{[t]_k}\|_F^2}{L^2}\right). \quad (8)$$

With these changes, the following theorem follows as Theorem 14.

Theorem 16. *Let $N_1, \ldots, N_t$ be a sequence of matrices with $N_{[t]} = [N_1, \ldots, N_t]$. Let $N_{[t]_k} = \hat{U}_{t_k}\hat{\Sigma}_{t_k}\hat{V}_{t_k}^\top$ be the best rank-k approximation to $N_{[t]}$. Let λ_i be the ith eigenvalue of $N_{[t]}N_{[t]}^\top$ and $L = \min_{i \neq j} |\lambda_i - \lambda_j| > 0$. Then for any unit vector $\mathbf{y} \in \mathbb{R}^m$, $\tilde{U}_{t_k}$ (generated by the Algorithm* DetUpdate*), under condition on ℓ from (8), satisfies:*

$$\begin{aligned}&\left|\min_{\mathbf{x}\in\mathbb{R}^k}\|\mathbf{y} - \hat{U}_{t_k}\mathbf{x}\| - \min_{\mathbf{x}\in\mathbb{R}^k}\|\mathbf{y} - \tilde{U}_{t_k}\mathbf{x}\|\right|\\ &\leq \frac{\sqrt{2}L}{\sqrt{L + 8\kappa^2\|N_{[t]}\|^2}\sqrt[4]{L^2 + 16\kappa^4\|N_{[t]}\|^4}}.\end{aligned}$$

Dataset	#Datapoints	#Features	% of Anomalies
Cod-RNA	488,565	8	33.33%
Protein-homology	145,751	74	0.89%
RCV1AD	100,274	1000	18.12%
Poker	1,025,010	10	7.63%
User-activity	129,328	83	10.69%

Table 1: Statistics of the experimental datasets.

The above theorem has an interpretation similar to that of Theorem 14. However, compared to Theorem 14 the requirement on ℓ is slightly weaker here.[8] This is because Algorithm DetUpdate computes the exact low-rank matrix at each timestep.

Remark 17. *The bounds on ℓ in (7) and (8) should be treated as existential results, as setting ℓ using these bounds are tricky. Practically, we noticed that setting $\ell \approx \sqrt{m}$ suffices to get good results for both Algorithms* DetUpdate *and* RandUpdate. *Another important point to remember is that both these algorithms can be used with any ℓ within $k \leq \ell \leq m$, the above bounds on ℓ are only to show theoretically that their performances are similar to using global singular value decomposition updates in Algorithm* AnomDetect.

5. EXPERIMENTAL ANALYSIS

In this section, we experimentally demonstrate that our proposed streaming approaches for anomaly detection easily adapt to unseen patterns arising in the stream and scale efficiently to big datasets. From now on, we refer to Algorithm AnomDetect with its singular vectors updated using either Algorithms GlobalUpdate, DetUpdate, or RandUpdate as **Global** , **ADeMS** , and **RandADeMS** respectively. As discussed earlier, Global is a baseline approach based on a standard idea. All our experimental evaluation were run on a machine with 2.5GHz Intel Core i7 processor and 16GB DDR3 SDRAM.

5.1 Experimental Setup

Datasets. We conducted experiments on datasets drawn from a diverse set of domains to demonstrate the wide applicability of our anomaly detection approach (see Table 1). *Cod-RNA* dataset consists of sequenced genomes, and the task here is to detect novel non-coding RNAs (ncRNAs) [38], which are labeled as anomalies. *Protein-homology* dataset is from the protein homology prediction task of the KDD Cup 2004 [7], and the task here is to predict which proteins in the database are homologous to a native (query) sequence. Non-homologous sequences are labeled as anomalies. *RCV1* dataset consists of a corpus of newswire stories (documents with only one label), grouped into categories [27]. In our evaluation, from all the categories, we used documents belonging to the 10 largest categories and the 30 smallest categories (labeled as anomalies). For features, we use a vocabulary of 1000 terms selected based on frequency. We refer to this modified *RCV1* dataset as *RCV1AD*. *Poker* dataset [29] consists of over $1,000,000$ instances and 10 attributes. Each record of this dataset is an example of a hand consisting of five playing cards drawn from a standard deck of 52. We labeled the largest two clusters as normal instances and all the else as anomalies. The *User-activity* is a non-public dataset that comes from an application that monitors employee network activity log for an enterprise. The goal here is to identify malicious employee actions (anomalies) that result in loss of intellectual property.

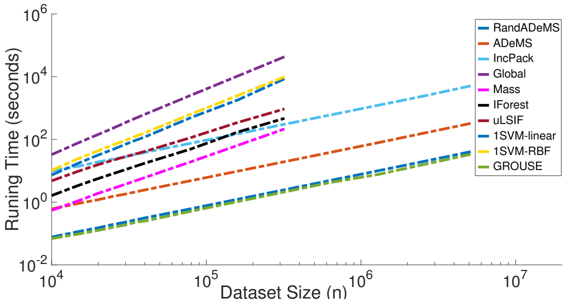


Figure 3: Scalability comparison of various algorithms.

Baselines. There are plenty of approaches for anomaly detection (as discussed in Section 2). We compared against seven popular algorithms (in addition to Global) for anomaly detection. These algorithms were chosen taking into account their scalability on large datasets. **1SVM-linear** and **1SVM-RBF** are one-class support vector machine classifiers with linear/radial-basis as kernel function. The output probability value of belonging to the anomalous class is treated as the anomaly score. We also compared against **IForest** [30], **Mass** [36], and Unconstrained Least-Squares Importance Fitting (**uLSIF**) [18] algorithms, which are all described in Section 2. These above five algorithms were chosen as our *non-incremental* baselines. As streaming and *incremental* competitors, we implemented two popular incremental Principal Component Analysis based schemes. One is the low-rank incremental approach (called **IncPack**) of Baker *et al.* [3] which unifies many previous approaches in this area. The other is an online subspace tracking algorithm (called **GROUSE**) of Balzano *et al.* [4] which is based on applying an incremental gradient descent technique on the Grassmannian manifold subspace. GROUSE is an online incremental algorithm that applies only simple updates at every timestep, therefore is also highly computationally efficient.

Parameter Settings. Except for IForest and Mass, all other competitors, including our proposed approaches, require an (initial) training set to bootstrap the process. As mentioned in Section 4, there are different ways this could be achieved. In our experiments, we assumed that there exists a small training set of non-anomalous samples. We set the size of the training set as 2000, and we draw these training samples randomly from the set of non-anomalous datapoints. Note that the training set size is much smaller compared to the actual dataset size. We also observed that our results are stable to variations in the training set (see results in Section 5.3).

After training, the number of datapoints (n_t's) given as input at each timestep is set to 5000 and as suggested in Remark 17 we set $\ell = \sqrt{m}$ where m is the feature size. We report effects of varying ℓ and n_t in Section 5.3. We set $k = m/5$. All non-incremental algorithms (1SVM-linear, 1SVM-RBF, IForest, Mass, and uLSIF) are considered to receive all the samples at once. The relevant parameters of these algorithms were tuned to obtain the best possible result.

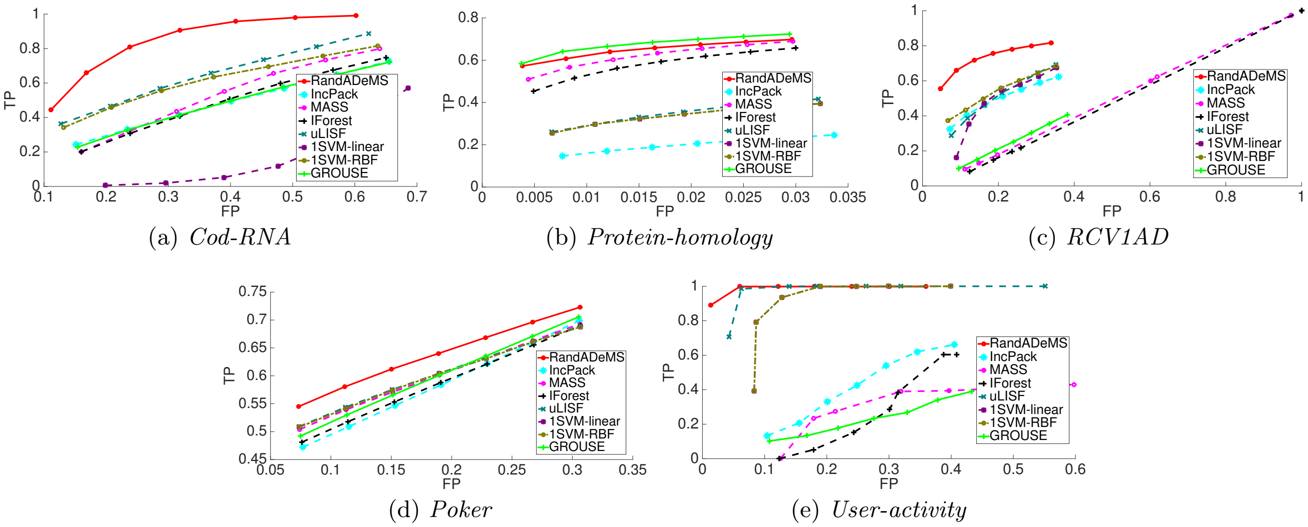
We used the standard evaluation metrics of True Positive rate (TP) and False Positive rate (FP). To generate these ROC curves, we use seven different threshold (ζ) numbers chosen based on the distribution of the anomaly scores.

5.2 Comparison between Different Algorithms

Figures 1 and 2 plot the ROC curves of the selected algorithms. Each point represents the average (TP and FP) of a 30-fold

[8] In fact, for small k's, and assuming $1 < \kappa \leq O(1)$ (implying $\Gamma_a = O(1)$ and $\Gamma_b = O(1)$), the bound on ℓ in (8) could be simplified to,

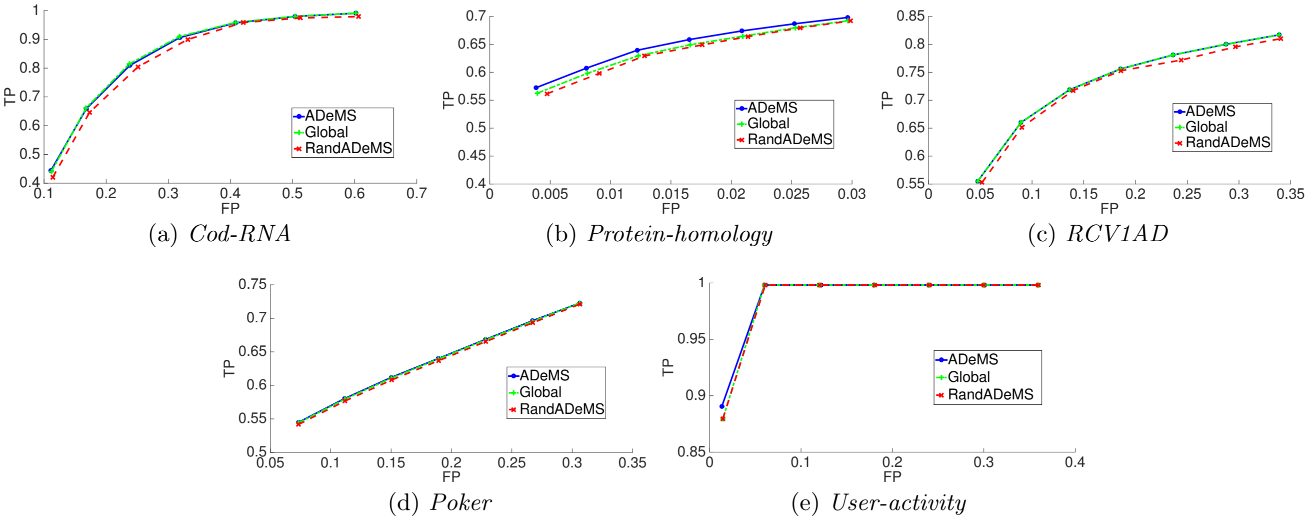
$$\ell = \Omega\left(\frac{\sqrt{m}\|N_{[t]}\|^2\|N_{[t]} - N_{[t]_k}\|_F^2}{L^2}\right).$$



(a) *Cod-RNA* (b) *Protein-homology* (c) *RCV1AD*

(d) *Poker* (e) *User-activity*

Figure 1: ROC curves for few compared approaches on various datasets.



(a) *Cod-RNA* (b) *Protein-homology* (c) *RCV1AD*

(d) *Poker* (e) *User-activity*

Figure 2: ROC curves for RandADeMS, ADeMS, and Global on various datasets.

cross-validation result, each time the training set was randomly selected from the normal samples and the order of samples was also randomly shuffled. We make the following observations:

1. From Figures 1(a), 1(b), 1(c), 1(d), and 1(e), it is evident that RandADeMS outperforms (dominates) other compared algorithms on most of the datasets, except on the *Protein-homology* dataset where GROUSE performs slightly better and on the *User-activity* dataset where there is a partial overlap between RandADeMS, 1SVM-RBF, and uLSIF. Note that the performance of RandADeMS is good, both when the fraction of anomalies is very high (such as in the *Cod-RNA* dataset, Figure 1(a)) or very small (such as in the *Protein-homology* dataset, Figure 1(b)). Other competitors demonstrate inconsistent performance across these datasets.

2. ADeMS, RandADeMS, and Global have very similar performances (Figures 2(a), 2(b), 2(c), 2(d), and 2(e)). It confirms our theoretical analysis (Theorems 14 and 16) that the Algorithms DetUpdate and RandUpdate provide a desired approximation to Algorithm GlobalUpdate. These figures also suggest that using a randomized low-rank SVD (as in RandADeMS), instead of the exact low-rank SVD (as in ADeMS) has little effect on the anomaly detection performance.

3. Figure 3 shows the scalability comparison (training + testing time) between the compared approaches. The datasets for this test were created by uniform down- and up-sampling the *Protein-homology* dataset, with feature size increased 5 times. RandADeMS is on average about 10+ times faster than ADeMS and on average about 100 times faster than IncPack. Its running time is almost identical to that of GROUSE, which as we mentioned earlier is a highly efficient online method for subspace tracking but with worse effectiveness (Figure 1). Compared with non-incremental competitors, RandADeMS is also faster than the efficient IForest and Mass anomaly detection algorithms, and is at least on average about 100 or more times faster than any of the other methods. In particular, RandADeMS and ADeMS, finish in couple of minutes, even when the dataset has few million instances.
Specifically, Table 5.2 lists the actual running comparison between ADeMS and RandADeMS on all the datasets. Notice that RandADeMS is notably more efficient than ADeMS (even ≈ 10 times more efficient on the *Protein-homology*

Dataset	ADeMS	RandADeMS
Cod-RNA	0.8221	0.2513
Protein-homology	3.1172	0.3384
RCV1AD	3.6559	3.2002
Poker	1.8791	0.4233
User-activity	0.2058	0.1593

Table 2: Actual running time comparison (in seconds) between ADeMS and RandADeMS.

dataset). The running times of both these algorithms are almost comparable for the *RCV1AD* and *User-activity* datasets, we believe that it is probably because both these datasets are *sparse*, i.e., percentage of non-zero entries in the input is small.

5.3 Stability Tests against Concept Drift and Parameters

Streaming algorithms are known to be sensitive to the order of data, or concept drift. To test the performance of our proposed RandADeMS in different concept drift scenarios, we used as input the *RCV1AD* dataset with its datapoints sorted by their actual timestamps and topic sizes (increasing and decreasing). The timestamp ordered data captures a realistic concept drift scenario where topics (news stories) arrive/fade over time, whereas by grouping by topics we simulate the scenarios where major concept drifts happen between topic transitions (when datapoints from one topic finish and next one starts). In Figure 4, we compared RandADeMS against a scheme (called **Non-Update**), where we use the same training setup as RandADeMS, but there are no updates to the U matrix over time (during testing). This captures a baseline algorithm which does not update for concept drift. Figure 4 shows that due to the singular vector updates, RandADeMS performs well even in the presence of major concept drifts (something that the Non-Update fails to do).

Figure 5(a) shows the performance of RandADeMS against different ℓ's on the *Protein-homology* dataset. The ℓ here ranges from 10 to 70 with increments of 10. The results show that even small values of ℓ get good anomaly detection performance. In Figure 5(b), we show the performance of RandADeMS on the *Poker* dataset across different batch sizes (n_t's). We ran the algorithm with batch sizes ranging from 1000 to $10,000$ with increments of 1000. These results indicate a very stable behavior of RandADeMS across different batch sizes. In Figure 5(c), we plot 30 different ROC curves for RandADeMS for different training initializations on the *RCV1AD* dataset. Each training set has 2000 samples randomly drawn from the set of non-anomalous datapoints. The points on the curves are within 5% of the averaged curve plotted in Figure 2(c), which demonstrates that the performance of RandADeMS holds independent of the training set used for bootstrapping.

Similar stable behavior was also observed for ADeMS (omitted here).

6. CONCLUSION

We proposed new deterministic and randomized sketching-based approaches to efficiently and effectively detect anomalies in large data streams. The resulting algorithms consume limited memory and require just one pass over the data. Our theoretical results show that these algorithms perform comparably with a global approach while being significantly faster and more memory efficient. Empirical evaluations on a variety of datasets illustrate the effectiveness and efficiency of the proposed approaches.

7. REFERENCES

[1] C. Aggarwal. *Outlier Analysis*. Springer, 2013.

[2] T. Ahmed, M. Coates, and A. Lakhina. Multivariate Online Anomaly Detection using Kernel Recursive Least Squares. In *INFOCOM*, 2007.

[3] C. G. Baker, K. A. Gallivan, and P. V. Dooren. Low-Rank Incremental Methods for Computing Dominant Singular Subspaces. *Linear Algebra and its Applications*, 2012.

[4] L. Balzano, R. Nowak, and B. Recht. Online Identification and Tracking of Subspaces from Highly Incomplete Information. In *Annual Allerton Conference*, 2010.

[5] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying Density-based Local Outliers. *SIGMOD*, 29(2), 2000.

[6] P. Businger. Updating a Singular Value Decomposition. *BIT*, 10(3):376–385, 1970.

[7] R. Caruana, T. Joachims, and L. Backstrom. KDD-Cup 2004: Results and Analysis. *JMLR*, 6(2):95–108, 2004.

[8] Y. Chahlaoui, K. Gallivan, and P. Van Dooren. Recursive Calculation of Dominant Singular Subspaces. *SIMAX*, 25(2), 2003.

[9] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 2009.

[10] X. Chen, W. Li, and W. Xu. Perturbation Analysis of the Eigenvector Matrix and Singular Vector Matrices. *Taiwanese Journal of Mathematics*, 16(1), 2012.

[11] M. Gabel, A. Schuster, and D. Keren. Communication-efficient Distributed Variance Monitoring and Outlier Detection for Multivariate Time Series. In *IPDPS*, 2014.

[12] M. Ghashami, A. Desai, and J. M. Phillips. Improved Practical Matrix Sketching with Guarantees. In *ESA 2014*.

[13] M. Ghashami, E. Liberty, J. M. Phillips, and D. P. Woodruff. Frequent Directions : Simple and Deterministic Matrix Sketching. *CoRR*, abs/1501.01711, 2015.

[14] M. Ghashami and J. M. Phillips. Relative Errors for Deterministic Low-Rank Matrix Approximations. In *SODA*, pages 707–717, 2014.

[15] G. H. Golub and C. F. Van Loan. *Matrix Computations*, volume 3. JHU Press, 2012.

[16] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Review*, 53(2), 2011.

[17] P. M. Hall, A. D. Marshall, and R. R. Martin. Incremental Eigenanalysis for Classification. In *BMVC*, 1998.

[18] S. Hido, Y. Tsuboi, H. Kashima, M. Sugiyama, and T. Kanamori. Statistical Outlier Detection using Direct Density Ratio Estimation. *KAIS*, 26(2), 2011.

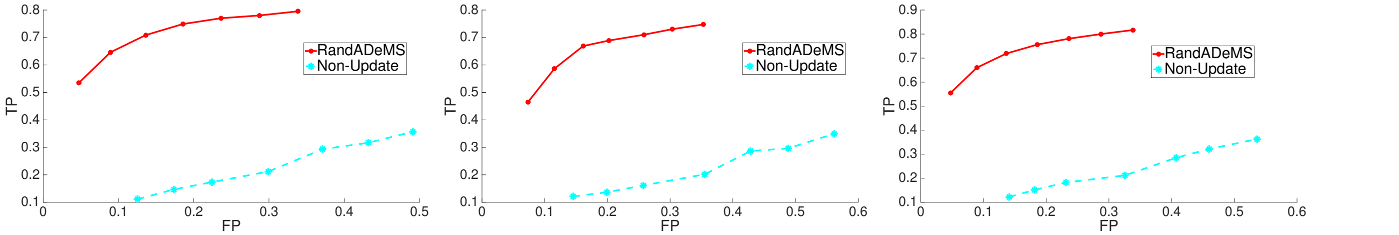
[19] H. Huang, H. Qin, S. Yoo, and D. Yu. Local anomaly descriptor: a robust unsupervised algorithm for anomaly detection based on diffusion space. *CIKM*, 2012.

[20] H. Huang, H. Qin, S. Yoo, and D. Yu. A new anomaly detection algorithm based on quantum mechanics. *ICDM*, 2012.

[21] H. Huang, H. Qin, S. Yoo, and D. Yu. Physics-Based Anomaly Detection Defined on Manifold Space. *TKDD*, 9(2), 2014.

[22] L. Huang, X. Nguyen, M. Garofalakis, J. M. Hellerstein, M. I. Jordan, A. D. Joseph, and N. Taft. Communication-efficient Online Detection of Network-wide Anomalies. In *INFOCOM*, 2007.

[23] L. Huang, X. Nguyen, M. Garofalakis, M. I. Jordan, A. Joseph, and N. Taft. In-network PCA and Anomaly Detection. In *NIPS*, pages 617–624, 2006.

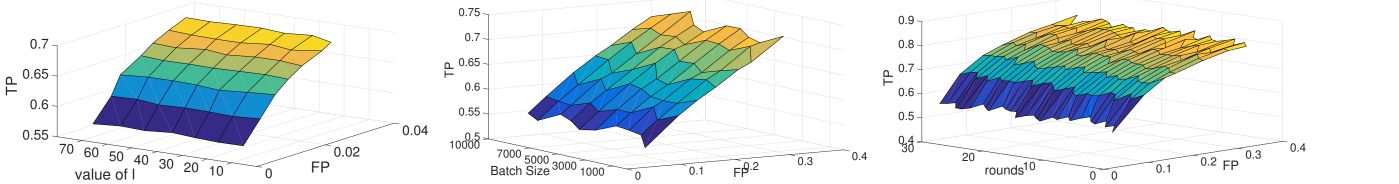


(a) Ordered by true datapoint timestamps

(b) Grouped by topics, ordered by increasing topic sizes

(c) Grouped by topics, ordered by decreasing topic sizes

Figure 4: Concept drift tests on different ordered *RCV1AD* data streams. The results show that RandADeMS provides a good and stable performance even in presence of inherent concept drift in the data stream.



(a) RandADeMS: Effect of varying ℓ

(b) RandADeMS: Stability against batchsizes

(c) RandADeMS: Stability against training set initializations

Figure 5: Figures 5(a) shows that even a small ℓ value suffices to get anomaly detection performance. Figures 5(b) and 5(c) show RandADeMS is stable against batch sizes (n_t's) and training set initializations, respectively.

[24] S. Kasiviswanathan, H. Wang, A. Banerjee, and P. Melville. Online L_1-Dictionary Learning with Application to Novel Document Detection. *NIPS*, 2012.

[25] A. Lakhina, M. Crovella, and C. Diot. Characterization of Network-wide Anomalies in Traffic Flows. In *SIGCOMM*, 2004.

[26] A. Levey and M. Lindenbaum. Sequential Karhunen-Loeve Basis Extraction and its Application to Images. *IEEE TIP*, 9(8), 2000.

[27] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A New Benchmark Collection for Text Categorization Research. *ACM SIGKDD Explorations Newsletter*, 5, 2004.

[28] E. Liberty. Simple and Deterministic Matrix Sketching. In *ACM SIGKDD*, pages 581–588, 2013.

[29] M. Lichman. UCI machine learning repository, 2013.

[30] F. T. Liu, K. M. Ting, and Z. H. Zhou. Isolation Forest. *IEEE ICDM*, pages 413–422, 2008.

[31] S. Liu, M. Yamada, N. Collier, and M. Sugiyama. Change-point Detection in Time-series Data by Relative Density-ratio Estimation. *Neural Networks*, 43:72–83, 2013.

[32] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online Learning for Matrix Factorization and Sparse Coding. *JMLR*, 11, 2010.

[33] M. Markou and S. Singh. Novelty Detection: A Review–part 1: Statistical Approaches. *Signal Processing*, 83(12), 2003.

[34] J. Misra and D. Gries. Finding Repeated Elements. *Science of Computer Programming*, 2(2):143–152, 1982.

[35] S. Papadimitriou, J. Sun, and C. Faloutsos. Streaming Pattern Discovery in Multiple Time-series. In *VLDB*, 2005.

[36] K. M. Ting, G. T. Zhou, F. T. Liu, and J. S. Tan. Mass Estimation and its Applications. *ACM SIGKDD*, 2010.

[37] A. Tsymbal. The Problem of Concept Drift: Definitions and Related Work. *Tech Report*, 2004.

[38] A. V. Uzilov, J. M. Keegan, and D. H. Mathews. Detection of Non-coding RNAs on the Basis of Predicted Secondary Structure Formation Free Energy Change. *BMC Bioinformatics*, 2006.

APPENDIX

A. FREQUENT DIRECTIONS ALGORITHM

Algorithm 5 presents the *Frequent Directions* algorithm [28, 13]. The algorithm operates in the column update model where the columns of the input matrix are added incrementally. The main idea behind the algorithm is that it periodically shrinks the ℓ orthogonal vectors by roughly the same amount. This is reminiscent to the counter shrinking idea used by Misra and Gries [34] for the problem of maintaining accurate frequency counts over a stream.

```
Algorithm 5: FrequentDirections
  Input: $Z \in \mathbb{R}^{m \times n}$, $S \in \mathbb{R}^{m \times \ell}$
  for each column $\mathbf{z}_i \in Z$ do
      $T \leftarrow [\mathbf{s}_1, \dots, \mathbf{s}_{\ell-1}, \mathbf{z}_i]$   (where $S = [\mathbf{s}_1, \dots, \mathbf{s}_\ell]$)
      $U \Sigma V^\top \leftarrow \textsc{Svd}(T)$  (where $\Sigma = \mathrm{diag}(\tilde{\sigma}_{t_1}, \dots, \tilde{\sigma}_{t_\ell})$)
      $\Sigma_t^{(\text{trunc})} \leftarrow \mathrm{diag}\left(\sqrt{\tilde{\sigma}_{t_1}^2 - \tilde{\sigma}_{t_\ell}^2}, \dots, \sqrt{\tilde{\sigma}_{t_{\ell-1}}^2 - \tilde{\sigma}_{t_\ell}^2}, 0\right)$
      $S \leftarrow U \Sigma_t^{(\text{trunc})}$
  end
  Return: $S$ and $U$
```

The matrix S is the matrix sketch. In our problem setup, at time t, if we have a sketch for $N_{[t-1]}$ and we want to create a sketch for $N_{[t]} = [N_{[t-1]}, N_t]$, then we could do so by passing N_t (as Z) and the sketch for $N_{[t-1]}$ (as S) in Algorithm FrequentDirections. However, as explained in Section 4.3, Algorithms DetUpdate and RandUpdate achieve the same (or similar) result in a much more efficient manner.