Data Stream Anomaly Detection through Principal Subspace Tracking

Pedro Henriques dos Santos Teixeira
pedro@intelie.com.br

Ruy Luiz Milidiú
milidiu@inf.puc-rio.br

Pontifícia Universidade Católica
R. M. de S. Vicente, 225
Rio de Janeiro, RJ 22453-900, Brazil

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning

Keywords

Unsupervised Learning, Data Streams, Anomaly Detection, Dimensionality Reduction, Time Series

ABSTRACT

We consider the problem of anomaly detection in multiple co-evolving data streams. In this paper, we introduce FRAHST (Fast Rank-Adaptive row-Householder Subspace Tracking). It automatically learns the principal subspace from N numerical data streams and an anomaly is indicated by a change in the number of latent variables. Our technique provides state-of-the-art estimates for the subspace basis and has a true dominant complexity of only $5Nr$ operations while satisfying all desirable *streaming* constraints. FRAHST successfully detects subtle anomalous patterns and when compared against four other anomaly detection techniques, it is the *only* with a consistent $F_1 \geq 80\%$ in the Abilene datasets as well as in the ISP datasets introduced in this work.

1. INTRODUCTION

The paper is organized as follows. We first present the problem and its motivation. Section 2 reviews the necessary literature to enable the narrative over the proposed FRAHST algorithm, which is then presented in section 3. Section 4 discusses experimental studies that demonstrate the effectiveness of our approach. Finally, we conclude in section 5 with dicussions and future research direction.

1.1 Problem motivation

Anomaly detection is an important and challenging problem that has been treated within diverse research areas. Our target domain is a data center, where huge volume of real-time data needs to be monitored by operations team with the objective to maintain high-availability and quality of the services.

The complexity of data centers poses many challenges for system administrators, who must constantly monitor their infrastructure to provide appropriate quality of service to their users. Supervisory processes are fundamental when running large systems and critical operations that need to be resilient to faults. Downtime can directly affect a company's income and certainly affect its reputation. Our goal is to devise a system that can detect anomalies as soon as possible in the infrastructure in order to minimize losses. Alarms should increase the situational awareness of human experts who can check the system and promptly act when needed and are spared from the rather implausible burden of continuous monitoring.

1.2 Data streams

Our work focus on measurements that are collected from data centers and naturally exhibit local correlations such as measurements from machines in the same cluster and from routers with redundant network links.

In this context, massive amounts of data are produced and it is not usually feasible to store all the data. The streaming data model may be viewed as a generalization of the traditional data warehouse model when the dataset size grows to infinity. This work focus on numerical values since they are ubiquitous to all monitored entities and a stream is considered to be an open-ended multivariate time series.

Definition 1. (Stream data model) The dataset $\mathcal{Z}$ is a growing sequence of N-dimensional vectors:

$$\mathcal{Z} \equiv \{\boldsymbol{z}(1), \boldsymbol{z}(2), \ldots, \boldsymbol{z}(t), \boldsymbol{z}(t+1), \ldots\}$$

with $\boldsymbol{z}(t) \in \Re^N$ for $t \geq 1$.

Researchers have started to redesign traditional mining algorithms to satisfy the requirements of the data streams context, where a algorithm must be incremental, work with one or few passes over the data and adapt to concept drifts. In the past few years, a new theory has emerged for reasoning about algorithms that work within these constraints on space, time and number of passes [1, 7, 19].

There are clear opportunities to combine machine learning algorithms with Data Stream Management Systems (DSMS) and leverage the processing of the incoming raw data.

1.3 Anomaly Detection

Anomaly detection refers to the problem of finding patterns in data that do not conform to expected behavior according to both local and temporal contexts.

In data centers, an anomaly is a short-lived deviation from its normal operation. Common source of anomalies are: software bugs (e.g. memory leaks, not optimized database queries), hardware malfunctioning (e.g. disk failures) or faults in the underlying subsystems (e.g. broken communication network access link).

Definition 2. (Anomaly detection) An anomaly α_i that starts at instant τ_i and lasts for ℓ_i intervals is represented as $\alpha_i = (\tau_i, \ell_i)$. Given the data so far, the task is to output a detection at interval d_i for every α_i such that $\tau_i \leq d_i \leq \tau_i + \ell_i$ and $(d_j - \tau_i)$ is minimal.

A recent survey [4] note that anomaly detection has traditionally dealt with record or transaction type data. They further indicate that most techniques require the entire test data before detecting anomalies, and it mentions very few online techniques. Indeed, most current algorithms assume the dataset fits in main memory [31]. Both aspects violate the requirement for real-time monitoring data streams.

It is a challenging task to detect failures in large dynamic systems because anomalous events may appear rarely and do not have fixed signature. The high dimensionality of the observation data, together with the frequent changes of system normal conditions resulting from user behavior as well as from changes in the infrastructure itself makes detection even more difficult.

In order to make the technique most widely applicable, an any-time *unsupervised* method that does not require training data is greatly desired as the nature of normal measurements can be learned and the method can adapt to variations in the structure of 'normality'. Because it is difficult to obtain training data, and we are more concerned with problems never seen before that notoriously arise in complex systems.

1.3.1 Evaluation metrics

A solution to this task is evaluated in terms of the detections considering the false alarms and missed anomalies. In real datasets, less than 5% of the data points correspond to anomalies, therefore the number of true negatives is very high and consequently the false positive rate is not a very meaningful measure. Hence, we choose the precision and recall measures as they give a more informative picture, and use their harmonic mean, the F_1 score, to represent both metrics.

1.4 Related work

Our work is mostly inspired by promising results using the SPIRIT algorithm [22], which associates anomalies to changes in the latent dimension by approximating the principal subspace incrementally. Using the same algorithm, InteMon [12] was evaluated in a data center at Carnegie Mellon but no metric or benchmark data was reported.

Lakhina et al. [15, 16] popularized using principal component analysis (PCA) for traffic anomaly detection where an anomaly is detected when the magnitude of the projection onto the anomalous subspace exceeds a Q-statistic threshold. This work requires the entire dataset in memory and an expensive SVD[1] computation, thus not appropriate for real-time applications.

[1]Singular Value Decomposition

Although an online formulation of the Q-PCA algorithm using a sliding window was suggested [17], it has been noted that using stale measurements based on a previous block of time to calculate the Q-statistic threshold resulted in a high number of false positives [2]. Moreover, [24] criticizes Lakhina's approach for being too sensitive to the number of principal components defining the normal subspace as a parameter and it points out that a large body of work used the same Abilene dataset, for which the parameters were highly optimized.

More recently, [2] address the streaming requirement and propose KOAD (Kernel-based Online Anomaly Detection). This work is an extended version of the Kernel Recursive Least Squares algorithm (KRLS). Since it is based on a regression method, the target variable is arbitrarily defined as the sum of the values in the input vector.

In [2], KOAD is described to identify a region of normality that corresponds to a high-density region of the space. [3] further formulates that the problem of learning such a representation consists in constructing a Minimum Volume Set (MVS). Therefore, the technique is evaluated against the One-Class Neighbor Machine (OCNM) algorithm proposed by [18] for estimating MVSs or density counter clusters, as these are known in the MVS literature. This algorithm is a block-based procedure that requires calculating the distance from every data point to every other in the dataset and similarly to Q-PCA will only be used for comparison purposes. KOAD is shown to have similar detection performance to the batch Q-PCA and OCNM approaches on the Abilene datasets.

1.5 Contributions

Our work points out that SPIRIT [22] is an extension to the PASTd algorithm [30] and thus inherits two major disadvantages that are widely known in the signal processing community: inability to provide orthonormal estimates and instability in the updating of the inverse power matrix that will 'explode' in the case of fading signals due to its use of the matrix inversion lemma. The loss of orthonormality in the tracked basis is critical because it is necessary by the rank estimation procedure in order to keep the reconstruction error respecting the algorithm's parameters. A re-orthonormalization step implies a computational complexity of $\mathcal{O}(Nr^2)$ per update and not $\mathcal{O}(Nr)$ as previously advertised. In the worst case analysis, even though unlikely, where $r = N$ this algorithm would offer no advantage over a batch SVD on the exponentially updated covariance matrix with exact subspace estimates. However, there is no known remedy for the numerical instability vulnerability.

We address those disadvantages and present a better algorithm. Namely, we introduce FRAHST (Fast Rank-Adaptive row-Householder Subspace Tracker) and the following are the primary contributions:

- It is the first rank-adaptive extension to the new the state-of-the-art recursive row-householder subspace tracking algorithm [26].
- It is numerically robust and stable.
- It reaches a dominant complexity of $\mathcal{O}(Nr)$ operations per interval which is the lower bound for an algorithm of this kind, which makes it more attractive for the streaming scenario.

Symbol	Description
N	Number of streams.
$\boldsymbol{z}(t)$	Data snapshot $\boldsymbol{z}(t) \equiv [z_1(t) \dots z_N(t)]^\top$ at time t.
α	Forgetting factor.
$\boldsymbol{\Phi}(t)$	Incrementally estimated covariance $N \times N$ matrix.
r	Number of latent variables.
$\boldsymbol{Q}(t)$	$N \times r$ projection matrix. The column vectors are the principal subspace basis.
$\boldsymbol{h}(t)$	$r \times 1$ vector of latent variables.
$\tilde{\boldsymbol{z}}$	$N \times r$ vector of the reconstruction of $\boldsymbol{z}(t)$.
E_t	Energy up to time t.
$\tilde{E}_t$	Energy captured by the latent model up to time t.
f_E, F_E	Lower and upper bounds on the fraction of total captured energy.
T	Total number of intervals so far.

Table 1: Description of notation.

- It is the only algorithm that consistently has a F_1 score equal or greater than 80% in all datasets.

This paper introduces the ISP datasets that were annotated by data center engineers at one of the largest[2] Internet Service Provider in South America with incidents that were not detected by the traditional monitoring systems. We show that embedding lagged data into the input vector is an effective modeling strategy and allows FRAHST to successfully capture temporal correlations and thus detect subtle anomalies in these datasets.

2. BACKGROUND

2.1 Principal subspace tracking

The main idea behind our proposed method is to perform dimensionality reduction while automatically adapting the number of latent variables. Our technique is reminiscent to principal component analysis, which is the optimum linear transform in the least square sense [14].

In the context of data streams, the entire data covariance matrix is not available at once and therefore the goal is the recursive estimation of the principal subspace of the time-recursively updated covariance matrix $\boldsymbol{\Phi}(t)$ of dimension $N \times N$,

$$\boldsymbol{\Phi}(t) = \alpha \boldsymbol{\Phi}(t-1) + \boldsymbol{z}(t)\boldsymbol{z}(t)^\top \qquad (1)$$

where $\boldsymbol{z}(t) \in \Re^N$ is the streaming data from Definition 1. The exponential forgetting factor $0 < \alpha < 1$ allows us to adapt to concept drifts over time.

A straightforward way to proceed is to apply SVD on the $\boldsymbol{\Phi}(t)$ at every step t. The eigenvalue decomposition can be expressed as follows:

$$\boldsymbol{\Phi}(t) = \boldsymbol{V}_r(t)\boldsymbol{\Lambda}_r\boldsymbol{V}_r^\top(t) + \boldsymbol{V}_{N-r}(t)\boldsymbol{\Lambda}_{N-r}\boldsymbol{V}_{N-r}^\top(t) \qquad (2)$$

We are interested in the dominant part of this decomposition, which is obtained by the dominant eigenvectors $\boldsymbol{V}_r(t)$ that span the principal subspace of rank r. The direct SVD approach typically requires $\mathcal{O}(N^3)$ operations but approximated schemes were devised to require much less operations.

Since usually $r \ll N$, the subspace tracking algorithms can be classified with respect to their computational complexity: methods requiring $\mathcal{O}(N^2 r)$ or $\mathcal{O}(N^2)$ will be classified as *high* complexity; $\mathcal{O}(Nr^2)$ as *medium* complexity and finally those with $\mathcal{O}(Nr)$ as *low* complexity. The algorithms in the last class are called *fast subspace trackers* and they are most suitable for real-time computing. The article of [5] constitutes a review of the results up to 1990, treating the first two classes, since the last class was not available at the time. The most complete reviews for the fast subspace trackers are available from [6] and [26].

2.2 Orthogonal iteration principle

Most (if not all) fast subspace trackers can be analyzed in terms of the orthogonal iteration, which is a generalization of the power method. The Owsley algorithm [21] was one of its first applications where a single orthogonal iteration is applied in each time step:

$$\boldsymbol{A}(t) = \boldsymbol{\Phi}(t)\boldsymbol{Q}(t-1) \qquad (3)$$

$$\boldsymbol{A}(t) = \boldsymbol{Q}(t)\boldsymbol{S}(t) \quad : \text{orthonormal factorization} \qquad (4)$$

where $\boldsymbol{A}(t)$ is an auxiliary matrix $N \times r$. Although Owsley used a QR-factorization in (4), Strobach [26] notes that if only a basis of the principal subspace is sought, then the shape of the square $r \times r$ S-matrix can be left completely undetermined. This allow most methods to move from $\mathcal{O}(Nr^2)$ to $\mathcal{O}(Nr)$, but the tracked basis vectors in $\boldsymbol{Q}(t)$ will no longer be the tracked eigenvectors. That is why these methods are more precisely defined as principal subspace trackers (i.e. rather than eigenspace trackers). This is not a disadvantage, because in most applications one is only interested in tracking the projection matrix, namely $\boldsymbol{Q}(t)\boldsymbol{Q}(t)^\top \approx \boldsymbol{V}(t)\boldsymbol{V}(t)^\top$.

Besides relaxing the constraint on the S-matrix, fast methods seek the direct updating of the orthonormal factorization of the A-matrix to avoid both an explicit covariance matrix and an orthogonalization step. The simplified recurrence from the model introduced in [25, 26]

$$\boldsymbol{Q}(t)\boldsymbol{S}(t) = \alpha\boldsymbol{Q}(t-1)\boldsymbol{S}(t-1) + \boldsymbol{z}(t)\underbrace{\boldsymbol{Q}(t-1)^\top\boldsymbol{z}(t)}_{\boldsymbol{h}(t)} \qquad (5)$$

underlies both subspace trackers of interest, and a brief introduction follows in the next two subsections. The vector $\boldsymbol{h}(t)$ correspond to the latent variables, which are analogous to the principal components in PCA.

2.3 PAST

The Projection Approximation Subspace Tracking algorithm, originally proposed in [30], is probably the best known approach for tracking the principal subspace and was originally derived by minimizing the PCA criterion, expressed in terms of the model's reconstruction square error.

PAST is derived in [26] where $\boldsymbol{Q}(t-1)^\top$ is pre-multiplied in (5) to obtain $\boldsymbol{S}(t) = \alpha\boldsymbol{S}(t-1) + \boldsymbol{h}(t)\boldsymbol{h}^\top(t)$. Therefore, PAST assumes *a priori* $\boldsymbol{Q}(t-1)^\top\boldsymbol{Q}(t) = \boldsymbol{I}$ projection approximation, which explains the widely acknowledged loss of orthonormality of the projection matrix in PAST as it violates the subspace propagation model. The algorithm follows after defining $\boldsymbol{P}(t) = \boldsymbol{S}^{-1}(t)$ and applying the matrix inversion lemma very much like classical recursive least squares (RLS) [11]. As a consequence, PAST is shown to collapse with overflow errors in [27] under a very noisy scenario.

PASTD is presented as a variant of the PAST algorithm in the same influential article [30], using the deflation technique. Yang highlights that the main update step in the

[2]Over 1,200 servers and 40,000 numerical charts generated continuously.

PASTd is identical, except for the step size, to Oja's learning rule [20], which was designed for extracting the first principal component by means of a single linear unit neural network. PASTd has been extended in [29] to provide rank estimates using information theoretic criteria, while SPIRIT [22] extends PASTd to allow rank discovery using energy thresholding. They are nearly identical to PASTd, hence are classified as low complexity S^{-1} domain algorithms with the same computational complexity and same disadvantages.

2.4 Fast Householder Subspace Tracker

Despite decades of research on subspace trackers, the new row-Householder approach [26] has recently appeared in the literature and is the state-of-the-art for this problem. It follows similar shape from the classical LORAF3 [25] but is motivated to achieve a dominant complexity of $3Nr$ operations per update, which is the lower bound for a problem of this kind. This method provides excellent superior subspace estimates and guarantees that the subspace basis are orthonormal.

The Householder reflection is very common in batch algorithms such as in implementations of QR decomposition. Strobach's algorithm operates directly on the S-domain, that are generally preferable than S^{-1}-domain techniques [11]. The technique is purely reflection based and an updating of inverse matrices is not used.

3. FRAHST

In this section, we present our method.

3.1 Tracking the subspace basis

We extend the original Householder subspace tracker [26] by assuming *a posteriori* projection approximation ($\psi = 0$ in the original paper). The derivation of the fast row-Householder tracker algorithm follows from the orthogonal decomposition of the data vector:

$$\boldsymbol{z}(t) = Z^{1/2}(t)\bar{\boldsymbol{z}}_\perp(t) + \boldsymbol{Q}(t-1)\boldsymbol{h}(t) \tag{6}$$

where
$$\boldsymbol{z}_\perp(t) = \boldsymbol{z}(t) - \boldsymbol{Q}(t-1)\boldsymbol{h}(t) \tag{7}$$

$$Z(t) = \boldsymbol{z}_\perp^\top(t)\boldsymbol{z}_\perp(t) \tag{8}$$

$$\bar{\boldsymbol{z}}_\perp(t) = Z^{-1/2}(t)\boldsymbol{z}_\perp(t). \tag{9}$$

Substituting (6) into (5) yields the following updating expression:

$$\boldsymbol{Q}(t)\boldsymbol{S}(t) = \begin{bmatrix}\boldsymbol{Q}(t-1) & \bar{\boldsymbol{z}}_\perp(t)\end{bmatrix} \times \begin{bmatrix}\alpha\boldsymbol{S}(t-1) + \boldsymbol{h}(t)\boldsymbol{h}^\top(t) \\ Z^{1/2}(t)\boldsymbol{h}^\top(t)\end{bmatrix}. \tag{10}$$

Strobach proposes the Householder reflection to compress the energy in the augmented $(r+1)\times r$ S-matrix. The Householder reflection is given by $\boldsymbol{H} = \boldsymbol{I} - 2\underline{\boldsymbol{v}}\underline{\boldsymbol{v}}^\top$ where $\underline{\boldsymbol{v}} = \begin{bmatrix}\boldsymbol{v}\\ \varphi\end{bmatrix}$ represents the plane of reflection with $\|\underline{\boldsymbol{v}}\| = 1$ and $\boldsymbol{H}\boldsymbol{H} = \boldsymbol{I}$. The goal is to annihilate the row vector in the appended S-matrix as follows:

$$\begin{bmatrix}\boldsymbol{S}(t)\\ 0\ldots 0\end{bmatrix} = \boldsymbol{H}(t)\begin{bmatrix}\alpha\boldsymbol{S}(t-1) + \boldsymbol{h}(t)\boldsymbol{h}^\top(t) \\ Z^{1/2}(t)\boldsymbol{h}^\top(t)\end{bmatrix} \tag{11}$$

and the Q-matrix may be updated accordingly

$$\begin{bmatrix}\boldsymbol{Q}(t) & \boldsymbol{z}_q(t)\end{bmatrix} = \begin{bmatrix}\boldsymbol{Q}(t-1) & \bar{\boldsymbol{z}}_\perp(t)\end{bmatrix}\boldsymbol{H}(t). \tag{12}$$

The solution to this problem is $\boldsymbol{H}(t)$ with a bottom-row vector that belongs to the nullspace of the augmented S-matrix. The condition yields a system of linear r equations. Equation-array (13a)-(13j) in Figure 1 is a quasicode listing of the entire subspace tracking algorithm.

3.2 Tracking the subspace rank

Most subspace tracking algorithms have the dimensionality r of the principal subspace given as a parameter. Our proposed method automatically discovers the rank r, so that we maintain a high percentage f_E of the energy $E(t) = \sum_{t=1}^{T}\|\boldsymbol{z}(t)\|^2$. Energy thresholding is a common method to estimate the number of principal components [14] and is adopted in SPIRIT [22], and enable detection of unusual patterns in data center measurements [12]. We let the reconstruction of $\boldsymbol{z}(t)$ based on the previously learned projection matrix be defined as $\tilde{\boldsymbol{z}}(t) = \boldsymbol{Q}(t-1)\boldsymbol{h}(t)$ and we similarly have the energy $\tilde{E}(t)$ of the reconstruction $\tilde{\boldsymbol{z}}$. The rank estimation procedure is shown in equations (13k)-(13u) in Figure 1, where we incrementally estimate both energies with the same forgetting factor from (1). It has been show in [22] that the relative squared error of the reconstruction, under this routine, satisfies:

$$1 - F_E \le \frac{\sum_{t=1}^{T}\|\boldsymbol{z}(t) - \tilde{\boldsymbol{z}}(t)\|^2}{\sum_{t=1}^{T}\|\boldsymbol{z}(t)\|^2} \le 1 - f_E$$

where T is the number of data points read so far. Nevertheless, this lemma *only* follows from the orthonormality of the projection matrix $\boldsymbol{Q}(t)$, which is guaranteed by the recursive row-Householder method.

Since r may change over time, our algorithm needs to restructure the two internal Q and S matrices. In order to best preserve the properties of each matrix and maintain the quality of the tracked subspace, we devise two simple heuristics to restructure both matrices. When r is incremented, we append the normalized orthogonal error given by the current updated projection matrix. The vector $\boldsymbol{z}_\perp^\dagger(t)$ from (13n) servers as an instantaneous estimate for the new basis able to capture interesting information in the new direction. When r is decremented, the matrices are truncated as shown in (13s) and (13t) This is similar to ideas employed in [29].

3.3 Exception handling

The zero-input case requires the operation of the algorithm in 'idle mode', which enables the algorithm to handle cases of vanishing inputs. It can be seen in (13a) and (13b) that a null input will zero both $Z(t)$ and $\boldsymbol{h}(t)$. Hence no updating of the basis estimate in $\boldsymbol{Q}(t)$ is necessary because there is no innovation in a zero input. We add a check to test the condition $Z(t) < \sigma$ before (13c) to bypass all remaining computations for that step. The singularity threshold σ is machine and platform dependent but is a positive constant very close to zero.

Fortunately, our rank estimation routine will already guarantee that the S-matrix is not rank deficient, hence we do not need to handle special conditions elsewhere.

3.4 Achieving lower asymptotic complexity

The entire algorithm shown in Figure 1 has time complexity of $5Nr + \mathcal{O}(N + r^3)$ flops per update. The space complexity $\mathcal{O}(Nr)$ is similarly very low. Even though r is

```
parameters: $0 < f_E < F_E < 1$ and $0 < \alpha < 1$
$r(0) \leftarrow 1$
$\boldsymbol{Q}(0) \leftarrow$ random orthonormal
$\boldsymbol{S}(0) \leftarrow \sigma\boldsymbol{I}$, where $\sigma$ is a small positive constant
for $t = 1, 2, \ldots$
    read $\boldsymbol{z}(t) \in \Re^N$
    $\boldsymbol{h}(t) = \boldsymbol{Q}^\top(t-1)\boldsymbol{z}(t)$    (13a)
    $Z(t) = \boldsymbol{z}^\top(t)\boldsymbol{z}(t) - \boldsymbol{h}^\top(t)\boldsymbol{h}(t)$    (13b)
    $\boldsymbol{X}(t) = \alpha\boldsymbol{S}(t-1) + \boldsymbol{h}(t)\boldsymbol{h}^\top(t)$    (13c)
    $\boldsymbol{X}^\top(t)\boldsymbol{b}(t) = Z^{1/2}\boldsymbol{h}(t) \xrightarrow{\text{solve}} \boldsymbol{b}(t)$    (13d)
    $\varphi^2(t) = \frac{1}{2} + \frac{1}{\sqrt{4(\boldsymbol{b}^\top(t)\boldsymbol{b}(t)+1)}}$    (13e)
    $\delta(t) = \frac{\varphi(t)}{Z^{1/2}(t)}$    (13f)
    $\boldsymbol{v}(t) = \left[\frac{1-2\varphi^2(t)}{2\varphi(t)}\right]\boldsymbol{b}(t)$    (13g)
    $\boldsymbol{S}(t) = \boldsymbol{X}(t) - \frac{1}{\delta(t)}\boldsymbol{v}(t)\boldsymbol{h}^\top(t)$    (13h)
    $\boldsymbol{e}(t) = \delta(t)\boldsymbol{z}(t) - \boldsymbol{Q}(t-1)\left[\delta(t)\boldsymbol{h}(t) - \boldsymbol{v}(t)\right]$    (13i)
    $\boldsymbol{Q}(t) = \boldsymbol{Q}(t-1) - 2\boldsymbol{e}(t)\boldsymbol{v}^\top(t)$    (13j)
    $E(t) = \alpha E(t-1) + \|\boldsymbol{z}\|^2$    (13k)
    $\tilde{E}(t) = \alpha\tilde{E}(t-1) + \|\boldsymbol{h}\|^2$    (13l)
    if $\tilde{E}(t) < f_E E(t)$ and $r(t) < N$    (13m)
        $\boldsymbol{z}^\dagger_\perp(t) = \boldsymbol{z}(t) - \boldsymbol{Q}(t)\boldsymbol{Q}^\top(t)\boldsymbol{z}(t)$    (13n)
        $\boldsymbol{Q}(t+1) = \left[\boldsymbol{Q}(t) \quad \boldsymbol{z}^\dagger(t)/\|\boldsymbol{z}^\dagger(t)\|\right]$    (13o)
        $\boldsymbol{S}(t+1) = \begin{bmatrix}\boldsymbol{S}(t) & \boldsymbol{0} \\ \boldsymbol{0} & \|\boldsymbol{z}^\dagger_\perp(t)\|^2\end{bmatrix}$    (13p)
        $r(t+1) = r(t) + 1$    (13q)
    else if $\tilde{E}(t) > F_E E(t)$ and $r(t) > 1$    (13r)
        $\boldsymbol{Q}(t+1) =$ delete the last column of $\boldsymbol{Q}(t)$    (13s)
        $\boldsymbol{S}(t+1) =$ delete last row & column of $\boldsymbol{S}(t)$    (13t)
        $r(t+1) = r(t) - 1$    (13u)
```

Figure 1: FRAHST algorithm

usually very small, one can perceive the typical $\mathcal{O}(r^3)$ complexity for solving the systems of linear equations in step (13d) as a stumbling block.

We achieve excellent estimates and lower asymptotic computational complexity of $5Nr + \mathcal{O}(N + r^2)$ following one of the suggestion in [26]. In particular, we work with the QR decompositions of the X and S $r \times r$ square matrices and recognize that (13c) and (13h) are 'QR = QR + rank one' updates. The algorithm for such recurrent updates is widely known and require $\mathcal{O}(r^2)$ flops [8]. Once we have the QR factors of the $\boldsymbol{X}(t)$, it is straight-forward to solve the linear system from (13d) with complexity $\mathcal{O}(r^2)$ by using back-substitution.

Similarly, we account for the resizing of the QR factors of the S-matrix using four procedures based on Givens rotations from [9, Chapter 12.5] for both appending and deleting column and row vector of the original matrix according to steps (13p) and (13t) whilst maintaining the QR structure with quadratic complexity.

```
last $\leftarrow 0$
for $t = 1, 2, \ldots$
    FRAHST($t$)    (14)
    if $r(t) > r(t-1)$    (15)
        if $t >$ last $+ 1$:  raise an alarm at $t$    (16)
        last $\leftarrow t$    (17)
```

Figure 2: Anomaly detection routine

3.5 Real-time anomaly detection system

Our anomaly detection procedure follows from the rank-adaptive nature of FRAHST. We propose to raise alarms whenever there is an increase in rank, as shown in Figure 2. This is the same intuition in [22], where observed data that cannot be satisfactorily explained by the current model is considered anomalous, and a new variable is introduced to guarantee a predefined reconstruction error.

In Figure 2, the control variable `last` is used to suppress alarms from consecutive rank increments which are likely to be false.

3.5.1 Event-driven Architecture

We devise a event-driven architecture where all modules are *publishers* or *subscribers* to an event broker. The most important components in a data center can be monitored using the Simple Network Management Protocol (SNMP), which include routers, access servers, switches, bridges, hubs, temperature sensors and computer hosts.

We monitor streams as defined by continuous queries over the incoming raw data. More specifically, the user chooses N data streams to be monitored together and a query can be placed to join the corresponding underlying streams in order to produce the input vector $\boldsymbol{z}(t)$ for our algorithm at periodic uniform intervals – the join operator and the output rate of the query are features of the core stream processing engine. A open-source complex event processing engine[3] is used to manage the processing of the raw data streams. Our solution has been adopted in a real data center, where it has been shown to achieve high throughput and low latency.

4. EXPERIMENTS

We implement all evaluated algorithms in R[4] [13]. We implement the QR update functions from Section 3.4 by calling the publicly available Fortran routines [10]. The streaming scenario is simulated by iterating over each input vector.

The default parameters for FRAHST and SPIRIT are set to the same recommended values from the literature [11, 22] $\alpha = 0.96$ and $[f_E, F_E] = [0.96, 0.98]$.

4.1 Datasets

We summarize the main features of all datasets in Table 2. Both Chlorine and Motes datasets were used for evaluating the SPIRIT algorithm in [22], where the accuracy in terms of the relative reconstruction error was the only quantitative metric reported. We use the Artificial dataset

[3]More information at http://www.espertech.com/.
[4]R is a language and a software environment for computing.

Dataset	N	T	duration	anomalies
Artificial	32	10000	-	-
Chlorine	166	4310	15 days	-
Motes	48	7712	1 month	-
Abilene Packets	121	2010	7 days	11
Abilene Flows	121	2010	7 days	15
ISP Routers	60	896	3 days	3
ISP Servers	24	600	2 days	8

Table 2: Description of datasets.

proposed in [26] to evaluate the quality of the tracked projection matrix.

The Abilene dataset seems to be the only public annotated datasets for the multivariate (numeric) anomaly detection problem. The Abilene dataset was used to compare KOAD, Q-PCA and OCNM in [2] for the anomaly detection problem. In their work, evaluation on the detections are carried out ignoring the first 400 intervals, therefore we follow the same approach here so the results are compatible.

We introduce the ISP dataset, which is comprised of two distinct multivariate measurements collected in the data center of a large Internet Service Provider. The measurements were collected every 6 minutes via SNMP and operations team kindly provided us the following two set of files with annotated anomalies from real incidents that were not alarmed by the current monitoring solution based on naïve thresholding.

Routers contains statistics from the data center routers, which are connected to five telecommunication operators by redundant network links. The dataset contains the number of bits per second in each link for both directions of communication averaged at each interval. The measurements were taken from 16/04/2009 to 19/04/2009. There was a communication failure with one of the operators, which caused two of the links to malfunction. The large failure was preceded by smaller loss of connectivity.

Servers contains CPU (*idle, usr*) and memory (*used, available*) usage statistics from each of the six machines within a cluster serving a specific application. The data was collected from 12/06/2009 to 13/06/2009, and contains anomalous events due to an unoptimized application that gained sudden popularity on the Internet. There was unexpected heavy load in two of the machines which had to be balanced to lower the latencies for the end-users.

4.2 Tracked subspace evaluation

In Table 3, we show the relative reconstruction error is calculated for the datasets Chlorine, Motes and Artificial.

According to the parameters both algorithms should maintain a reconstruction error between 0.02 and 0.04 by definition. Nevertheless, it can be seen that SPIRIT *fails* to fulfill this requirement and the relative reconstruction errors are twice from the expected values in two of the datasets.

Dataset	FRAHST	r	SPIRIT	r
Chlorine	**0.0236**	1-3	0.0350	1-2
Motes	**0.0206**	4-6	0.0804	4-5
Artificial	**0.0347**	3-4	0.0899	3-4

Table 3: Relative squared reconstruction error for the rank adaptive algorithms. Parameters $[f_E, F_E] = [0.96, 0.98]$ imply that the error should be between 0.02 and 0.04.

Method	Abilene Packets	Abilene Flows	ISP Routers	ISP Servers
FRAHST	0.91	0.87	**0.80**	**0.86**
SPIRIT	0.50	0.74	0.40	0.46
KOAD	0.86	**0.93**	0.15	0.54
Q-PCA	**0.95**	0.89	0.67	0.71
OCNM	0.77	**0.93**	0.25	0.50

Table 4: F_1 score on the anomaly detection task.

4.3 Anomaly detection evaluation

We summarize the anomaly detection results in Table 4. We observe that Abilene anomalies are mostly 'spikes', which seems to benefit KOAD and OCNM. SPIRIT is less sensitive to broken correlations, since it allows a greater than expected reconstruction error. Surprisingly, KOAD had 21 false alarms in the ISP Routers. We illustrate the behavior of the different algorithms in Figure 3.

4.3.1 Considerations

The parameters for the algorithms KOAD, Q-PCA and OCNM are set based on their best performance in the Abilene datasets [2]. For the ISP Routers, we set KOAD's thresholds $[v_1, v_2] = [0.04, 0.08]$ to decrease false alarms but detect at least one anomaly and let $k = 10$ for OCNM otherwise the algorithm raise alarms for all points in the 'night period', since there is less traffic. For the ISP Servers, we use for KOAD the same parameters used in the Abilene Packets, while for Q-PCA $r = 2$ yielded the best results. In this dataset, FRAHST and SPIRIT perform better when setting $f_E = 0.97$.

4.3.2 Modeling strategy

Here, we present results for the ISP datasets exploring two important modeling techniques:

Centering the data implies a principal subspace technique become conceptually closer to incremental PCA, since the captured energy is the variance of the principal components. Under streaming constraints, the actual mean of the data is unknown but we can incrementally estimate the sample mean $\boldsymbol{\mu}(t)$ applying the same exponential forgetting factor as before:

$$\boldsymbol{\mu}(t) = \frac{t-1}{t}\alpha\boldsymbol{\mu}(t-1) + \frac{1}{t}\boldsymbol{z}(t). \qquad (18)$$

The input data vector for the algorithm is now $\hat{\boldsymbol{z}}(t) = \boldsymbol{z}(t) - \boldsymbol{\mu}(t)$.

Temporal correlations can be explicitly captured by enlarging the dimension of the input vector by concatenating the lagged l past values:

$$\boldsymbol{z}_l(t) = \begin{bmatrix}\boldsymbol{z}^T(t) & \boldsymbol{z}^T(t-1) & \dots & \boldsymbol{z}^T(t-l)\end{bmatrix}^\top \qquad (19)$$

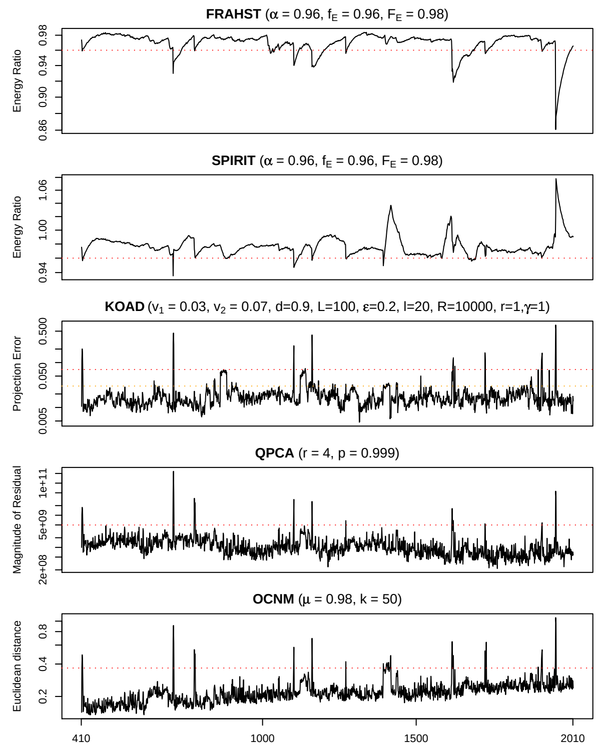

Figure 3: Determinant variables and thresholds (dotted lines) for all algorithms during the anomaly detection evaluation on the Abilene Packets dataset.

where $z_l(t) \in \Re^{lN}$. Embedding such auto-correlations allows our algorithm to learn a better model by working on a richer dimension.

We employ both techniques and $\hat{z}_l(t)$ becomes the input data vector for the algorithms. For a real-time system, the implementation of both techniques can be accomplished in conjunction with a DSMS[5]. We continued normalizing the incoming data only for the KOAD algorithm as suggested by the authors, because we notice that otherwise this RLS-based algorithm suffers from severe numerical instabilities and *fails* to produce correct results. RLS algorithms are notoriously unstable in a finite precision environment, and there was not much we could do otherwise.

For the experiments with the ISP Routers dataset, we set $l = 5$ which corresponds to a total lag of 30 minutes and an enhanced dimension of 893×300. And for the ISP Server dataset, we chose $l = 10$ which corresponds to a total lag of one hour and an enhanced dimension of 591×240. Both sizes of auto-correlation windows are adequate for monitoring these data streams and increasing this size did not improve the results for any of the algorithms.

5. CONCLUSIONS

We observed a few problems with the related methods:

[5]For example, in Esper, this can be accomplished using the *average* statistics view and the *previous* function.

SPIRIT does not guarantee expected accuracy. This is due to PASTd's inability to provide orthonormal estimates.

KOAD has too many false alarms in the ISP datasets and over 11 non-intuitive parameters. The RLS mechanism *fails* when normalization is not performed due to numerical instabilities.

Q-PCA requires the rank that defines normality as a parameter, which is *difficult* and not adequate to define.

OCNM is very expensive computationally and results vary greatly with the density function that is used.

In the light of these issues, we contribute FRAHST, a new rank-adaptive algorithm for fast principal subspace tracking with a true dominant complexity of $\mathcal{O}(Nr)$. The method is robust and captures sudden changes in the correlation structure of high-dimensional data.

We compared our technique with other two online and two batch algorithms for anomaly detection in four different datasets, and it achieved overall excellent performance being the only algorithm with $F_1 \geq 80\%$ in all experimented datasets. We showed how embedding lagged values in the input vector allows temporal correlations to be captured by a pre-processing step that can be easily performed online. It effectively allowed a subtle broken correlation to be detected corresponding to a serious failure caused by a telecommunication operator but which was not discovered by the traditional mechanisms when it occurred.

The results indicate that a robust subspace tracker is well suited for spotting anomalies in streaming data of low intrinsic dimension, even when compared to algorithms that can look at the entire dataset more than once. FRAHST was consistently better than SPIRIT in all criteria, hence we can safely recommend it for the same tasks SPIRIT has been used in the literature, such as forecasting.

A real-time system was successfully implemented to monitor the data center of a ISP and is a good use case for unsupervised anomaly detection in the industry.

Our work touches many important subjects such dimensionality reduction, rank estimation and anomaly detection under the streaming constraint and offers a useful *any-time* fast method for learning patterns in multiple streams of data.

5.1 Future work

Our algorithm depends on intuitive parameters: the forgetting factor α and desired reconstruction error $1 - f_E$. But we acknowledge that they might vary accordingly to the data, and it would be much better to have a totally parameter-free algorithm. Adaptive memory concepts from adaptive filtering might be applied; and new results from random matrix theory concerning the Tracy-Widom distribution seem to offer automatic ways to calculate optimal dynamic detection thresholds under clear formulations [23]. Under this probabilistic framework, a degree of confidence may be assigned to each alarm.

Another interesting direction of research is to extend the work of [31] to handle multivariate data in a streaming scenario, which would certainly contribute to the set of available techniques for the task considered in this paper.

6. REFERENCES

[1] C. Aggarwal. *Data Streams: Models and Algorithms.* Advances in Database Systems. Springer, 2007.

[2] T. Ahmed, M. Coates, and A. Lakhina. Multivariate online anomaly detection using kernel recursive least squares. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications.*, pages 625–633. IEEE, 2007.

[3] T. Ahmed, B. Oreshkin, and M. Coates. Machine learning approaches to network anomaly detection. In *in Proceedings of the Second Workshop on Tackling Computer Systems Problems with Machine Learning (SysML*, 2007.

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

[5] P. Comon and G. H. Golub. Tracking a few extreme singular values and vectors in signal processing. *Proceedings of the IEEE*, 78(8):1327–1343, Aug 1990.

[6] X. G. Doukopoulos and G. V. Moustakides. Fast and stable subspace tracking. *IEEE Transactions on Signal Processing*, 56:1452–1465, apr 2008.

[7] J. Gama and M. M. Gaber. *Learning From Data Streams : Processing Techniques in Sensor Networks.* Springer, September 2007.

[8] P. E. Gill, G. H. Golub, W. A. Murray, and M. A. Saunders. Methods for modifying matrix factorizations. Technical report, Stanford University, Stanford, CA, USA, 1972.

[9] G. H. Golub and C. F. Van Loan. *Matrix computations (3rd ed.).* Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[10] J. Hajek. qrupdate: a fortran library for fast updates of qr and cholesky decompositions. http://sourceforge.net/projects/qrupdate/, 2009.

[11] S. Haykin. *Adaptive filter theory (3rd ed.).* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[12] E. Hoke, J. Sun, J. D. Strunk, G. R. Ganger, and C. Faloutsos. Intemon: continuous mining of sensor data in large-scale self-infrastructures. *SIGOPS Oper. Syst. Rev.*, 40(3):38–44, 2006.

[13] R. Ihaka and R. Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.

[14] I. T. Jolliffe. *Principal Component Analysis.* Springer Series in Statistics. Springer, 2002.

[15] A. Lakhina, M. Crovella, and C. Diot. Diagnosing network-wide traffic anomalies. *SIGCOMM Comput. Commun. Rev.*, 34(4):219–230, 2004.

[16] A. Lakhina, M. Crovella, and C. Diot. Mining anomalies using traffic feature distributions. In *SIGCOMM âĂŹ05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 217–228, New York, NY, USA, 2005. ACM.

[17] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. Kolaczyk, and N. Taft. Structural analysis of network traffic flows, 2004.

[18] A. Munoz and J. M. Moguerza. Estimation of high-density regions using one-class neighbor machines. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(3):476–480, 2006.

[19] S. Muthukrishnan. Data streams: algorithms and applications. *Found. Trends Theor. Comput. Sci.*, 1(2):117–236, 2005.

[20] E. Oja, J. Karhunen, L. Wang, and R. Vigario. Principal and independent components in neural networks - recent developments. In *In Proc. VII Italian Workshop on Neural Nets*, pages 16–35, 1995.

[21] N. Owsley. Adaptive data orthogonalization. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP âĂŹ78.*, volume 3, pages 109–112, Apr 1978.

[22] S. Papadimitriou, J. Sun, and C. Faloutsos. Streaming pattern discovery in multiple time-series. In *Proceedings of the 31st international conference on Very large data bases*, pages 697–708. VLDB Endowment, 2005.

[23] P. O. Perry and P. J. Wolfe. Minimax rank estimation for subspace tracking. Submitted, January 2009.

[24] H. Ringberg, A. Soule, J. Rexford, and C. Diot. Sensitivity of pca for traffic anomaly detection. *SIGMETRICS Perform. Eval. Rev.*, 35(1):109–120, 2007.

[25] P. Strobach. Low-rank adaptive filters. *Signal Processing, IEEE Transactions on*, 44(12):2932–2947, Dec 1996.

[26] P. Strobach. The fast recursive row-householder subspace tracking algorithm. *Signal Processing*, In Press, Corrected Proof, 2009.

[27] P. Strobach. The householder compressor theorem and its application in subspace tracking. *Signal Processing*, 89(5):857–875, 2009.

[28] P. H. S. Teixeira. Data stream anomaly detection through principal subspace tracking. Master's thesis, Pontifícia Universidade Católica, Rio de Janeiro, Brazil, September 2009.

[29] B. Yang. An extension of the pastd algorithm to both rank and subspace tracking. *Signal Processing Letters, IEEE*, 2(9):179–182, Sep 1995.

[30] B. Yang. Projection approximation subspace tracking. *Signal Processing, IEEE Transactions on*, 43(1):95–107, Jan 1995.

[31] D. Yankov, E. Keogh, and U. Rebbapragada. Disk aware discord discovery: finding unusual time series in terabyte sized datasets. *Knowl. Inf. Syst.*, 17(2):241–262, 2008.