

Changepoint detection for high-dimensional time series with missing data

Yao Xie* , Jiaji Huang* , Rebecca Willett *

August 30, 2012

Abstract

This paper describes a novel approach to changepoint detection when the observed high-dimensional data may have missing elements. The performance of classical methods for changepoint detection typically scales poorly with the dimensionality of the data, so that a large number of observations are collected after the true changepoint before it can be reliably detected. Furthermore, missing components in the observed data handicap conventional approaches. The proposed method addresses these challenges by modeling the dynamic distribution underlying the data as lying close to a time-varying low-dimensional submanifold embedded within the ambient observation space. Specifically, streaming data is used to track a submanifold approximation, measure deviations from this approximation, and calculate a series of statistics of the deviations for determining when the underlying manifold has changed in a sharp or unexpected manner. The approach described in this paper leverages several recent results in the field of high-dimensional data analysis, including subspace tracking with missing data, multiscale analysis techniques for point clouds, online optimization, and changepoint detection performance analysis. Simulations and experiments highlight the robustness and efficacy of the proposed approach in detecting an abrupt change in an otherwise slowly-varying low-dimensional manifold.

1 Introduction

Changepoint detection is a form of anomaly detection where the anomalies of interest are abrupt temporal changes in a stochastic process [1, 2]. A good changepoint detection algorithm will accept a sequence of random variables whose distribution may change abruptly at one time, detect such a change as soon as possible, and also have long period between false detections. In many modern applications, the stochastic process is non-stationary away from the changepoints and very high dimensional, resulting in significant statistical and computational challenges. For instance, we may wish to quickly identify changes in network traffic patterns [3], social network interactions [4], surveillance video [5], or solar flare imagery [6, 7].

Traditional changepoint detection methods typically deal with a sequence of low-dimensional, often scalar, random variables. Naïvely applying these approaches to high-dimensional data is impractical because the underlying high-dimensional distribution cannot be accurately estimated and used for developing test statistics. This results in detection delays and false alarm rates that scale poorly with the dimensionality of the problem. Thus the primary challenge here is to develop a rigorous method for extracting meaningful low-dimensional statistics from the high-dimensional data stream without making restrictive modeling assumptions.

Our method addresses these challenges by using multiscale online manifold learning to extract univariate changepoint detection test statistics from high-dimensional data. We model the dynamic distribution underlying the data as lying close to a time-varying, low-dimensional submanifold embedded within the ambient observation space. This submanifold model, while non-parametric, allows us to generate meaningful test statistics for robust and reliable changepoint detection, and the multiscale structure allows for fast, memory-efficient computations. Furthermore, these statistics can be calculated even when elements are missing from the observation vector.

While manifold learning has received significant attention in the machine learning literature [8–14], online learning of a dynamic manifold remains a significant challenge, both algorithmically and statistically. Most existing methods are “batch”, in that they are designed to process a collection of independent observations all lying near the same static submanifold, and all data is available for processing simultaneously.

In contrast, our interest lies with “online” algorithms, which accept streaming data and sequentially update an estimate of the underlying dynamic submanifold structure, and changepoint detection methods which identify

*Yao Xie (email: yao.c.xie@gmail.com), Jiaji Huang (email: jiaji.huang@duke.edu), and Rebecca Willett (email: willett@duke.edu) are with the Department of Electrical and Computer Engineering at the Duke University. This work was supported by DARPA award # FA8650-11-1-7150, AFOSR award # FA9550-10-1-0390, AFOSR award # FA9550-11-1-0028, and NSF CAREER award # NSF-CCF-06-43947.

significant changes in the submanifold structure rapidly and reliably. Recent progress for a very special case of submanifolds appears in the context of subspace tracking. For example, the Grassmannian Rank-One Update Subspace Estimation (GROUSE) [15] and Parallel Estimation and Tracking by REcursive Least Squares (PETRELS) [16] [17] effectively track a single subspace using incomplete data vectors. The subspace model used in these methods, however, provides a poor fit to data sampled from a manifold with non-negligible curvature.

1.1 Related work

At its core, our method basically tracks a time-varying probability distribution underlying the observed data, and uses this distribution to generate statistics for effective changepoint detection. For sequential density estimation problems such as this, it is natural to consider an online kernel density estimation (KDE) method. A naive variant of online KDEs would be quite challenging in our setting, however, because if we model the density using a kernel at each observed data point, then the amount of memory and computation required increases linearly with time and is poorly suited to large-scale streaming data problems. Ad-hoc “compression” or “kernel herding” methods for online kernel density estimation address this challenge [18] but face computational hurdles. Furthermore, choosing the kernel bandwidth, and particularly allowing it to vary spatially and temporally, is a significant challenge. Recent works consider variable bandwidth selection using expert strategies which increase memory requirements [19, 20]. Some of these issues are addressed by the RODEO method [21], but the sparse additive model assumed in that work limits the applicability of the approach; our proposed method is applicable to much broader classes of high-dimensional densities. Finally, in high-dimensional settings asymmetric kernels which are not necessarily symmetric or coordinate-aligned appear essential for approximating densities on low-dimensional manifolds, but learning time-varying, spatially-varying, and anisotropic kernels remains an open problem. In a sense, our approach can be considered a memory-efficient *sparse* online kernel density estimation method, where we only track a small number of kernels, and we allow the number of kernels, the center of each kernel, and the shape of each kernel to adapt to new data over time.

Our approach also has close connections with Gaussian Mixture Models (GMMs) [22–24]. The basic idea here is to approximate a probability density with a mixture of Gaussian distributions, each with its own mean and covariance matrix. The number of mixture components is typically fixed, limiting the memory demands of the estimate, and online expectation-maximization algorithms can be used to track a time-varying density [25]. However, this approach faces several challenges in our setting. In particular, choosing the number of mixture components is challenging even in batch settings, and the issue is aggravated in online settings where the ideal number of mixture components may vary over time. Furthermore, without additional modeling assumptions, tracking the covariance matrices for each of the mixture components is very ill-posed in high-dimensional settings.

Our approach is closely related to Geometric Multi-Resolution Analysis (GMRA) [14], which was developed for analyzing intrinsically low-dimensional point clouds in high-dimensional spaces. The basic idea of GMRA is to first iteratively partition a dataset to form a multiscale collection of subsets of the data, then find a low-rank approximation for the data in each subset, and finally efficiently encode the difference between the low-rank approximations at different scales. This approach is a batch method without a straightforward extension to online settings.

1.2 Motivating applications

The proposed method is applicable in a wide variety of settings. Consider a video surveillance problem. Many modern sensors collect massive video streams which cannot be analyzed by human due to the sheer volume of data; for example, the ARGUS system developed by BAE Systems is reported to collect video-rate gigapixel imagery [26, 27], and the Solar Dynamics Observatory (SDO) collects huge quantities of solar motion imagery “in multiple wavelengths to [help solar physicists] link changes in the surface to interior changes” [28]. Solar flares have a close connection with geomagnetic storms, which can potentially cause large-scale power-grid failure. In recent years the sun has entered a phase of intense activity, which makes monitoring of solar flare bursts an even more important task [7]. With these issues in mind, it is clear that somehow *prioritizing* the available data for detailed analysis is an essential step in the timely analysis of such data. If we can reliably detect statistically significant changes in the video, we can focus analysts’ attention on salient aspects of the dynamic scene. For example, we may wish to detect a solar flare in a sequence of solar images in real time *without* an explicit model for flares, or detect anomalous behaviours in surveillance video [29]. Saliency detection has been tackled previously [30, 31], but most methods do not track gradual changes in the scene composition and do not detect *temporal* changepoints.

A second motivating example is credit history monitoring, where we are interested in monitoring the spending pattern of a user and raising an alarm if a user’s spending pattern is likely to result a default [32]. Here normal spending patterns may evolve over time, but we would expect a sharp change in the case of a stolen identity.

An additional potential application arises in computer network anomaly detection [33]. Malicious attacks or network failure can significantly affect the characteristics of a network [3, 34]. Recent work has shown that network traffic data is well-characterized using submanifold structure [35], and using such models may lead to more rapid detection of changepoints with fewer false alarms.

1.3 Contributions and paper organization

The primary contributions of this work are two-fold: we present (a) a fast method for online tracking of a dynamic submanifold underlying very high-dimensional noisy data with missing elements and (b) a principled changepoint detection method based on easily computed residuals of our online submanifold approximation. These methods are supported by both theoretical analyses and numerical experiments on simulated and real data.

The paper is organized as follows. In Section 2 we formally define our setting and problem. Section 3 describes our multiscale submanifold model and tracking algorithm, which is used to generate the statistics used in the changepoint detection component described in Section 4. Several theoretical aspects of the performance of our method are described in Section 5, and the performance is illustrated in several numerical examples in Section 6.

2 Problem Formulation

Suppose we are given a sequence of data x_1, x_2, \dots , and for $t = 1, 2, \dots$, $x_t \in \mathbb{R}^D$, where D denotes the *ambient dimension*. The data are noisy measurements of points lying on a submanifold $v_t \in \mathcal{M}_t$:

$$x_t = v_t + w_t. \quad (1)$$

The *intrinsic dimension* of the submanifold \mathcal{M}_t is d . We assume $d \ll D$. The noise w_t is a zero mean white Gaussian random vector with covariance matrix $\sigma^2 I$. The underlying submanifold \mathcal{M}_t may vary slowly with time. At each time t , we only observe a partial vector x_t at locations $\Omega_t \in \{1, \dots, D\}$. Let \mathcal{P}_{Ω_t} represent the $|\Omega_t| \times D$ matrix that selects the axes of \mathbb{R}^D indexed by Ω_t ; we observe $\mathcal{P}_{\Omega_t} x_t$.

Our goal is to design an online algorithm that generates a sequence of approximations $\widehat{\mathcal{M}}_t$ which tracks \mathcal{M}_t when it varies slowly, and detects anomalies as soon as possible when the submanifold changes abruptly. The premise is that the statistical properties of the tracking error will be different when the submanifold varies slowly versus when it changes abruptly.

Define the operator

$$\mathcal{P}_{\mathcal{M}} x_t = \arg \min_{x \in \mathcal{M}} \|x - x_t\|^2 \quad (2)$$

as the projection of observation x_t on to \mathcal{M} . If we had access to all the data simultaneously without any memory constraints, we might solve the following batch optimization problem using all data up to time t for an approximation:

$$\widehat{\mathcal{M}}_t^\circ \triangleq \arg \min_{\mathcal{M}} \left\{ \sum_{i=1}^t \alpha^{t-i} \|\mathcal{P}_{\Omega_i}(x_i - \mathcal{P}_{\mathcal{M}} x_i)\|^2 + \mu \text{pen}(\mathcal{M}) \right\}, \quad (3)$$

where $\|x\|$ denotes the Euclidean norm of a vector x , $\text{pen}(\mathcal{M})$ denotes a regularization term which penalizes the complexity of \mathcal{M} , $\alpha \in (0, 1]$ is a discounting factor on the approximation error at each time t , and μ is a user-determined constant that specifies the relative weights of the data fit and regularization terms.

Note that (3) cannot be solved without retaining all previous data in memory, which is impractical for the applications of interest. To address this, we instead consider an approximation to the cost function in (3) of the form $F(\mathcal{M}) + \mu \text{pen}(\mathcal{M})$. To develop an online algorithm, instead of solving (3), we find a sequence of approximations $\widehat{\mathcal{M}}_1, \dots, \widehat{\mathcal{M}}_t$ (without storing historic data), such that $\widehat{\mathcal{M}}_{t+1}$ is computed by updating the previous approximation $\widehat{\mathcal{M}}_t$ using $F(\mathcal{M})$ and the current datum x_{t+1} . In Section 3, we will present several forms of $F(\mathcal{M})$ that lead to recursive updates and efficient tracking algorithms. One example of a MOUSSE approximation is illustrated in Figure 1; the context is described in more detail in Section 6.2.

Given the sequence of submanifold estimates $\widehat{\mathcal{M}}_1, \dots, \widehat{\mathcal{M}}_t$, we can compute the distance of each x_t to $\widehat{\mathcal{M}}_t$, which we denote $\{e_t\}$. We then apply changepoint detection methods to the sequence of tracking errors $\{e_t\}$. In particular, we assume that when there is no anomaly, the e_t are i.i.d. with distribution ν_0 . When there is an anomaly, there exists an unknown time $\kappa < t$ such that before the changepoint e_1, \dots, e_κ are i.i.d. with distribution ν_0 , and after the changepoint, $e_{\kappa+1}, \dots$ are i.i.d. with distribution ν_1 . Our goal is to detect the anomaly as quickly as possible after it occurs, and make as few false alarms as possible.

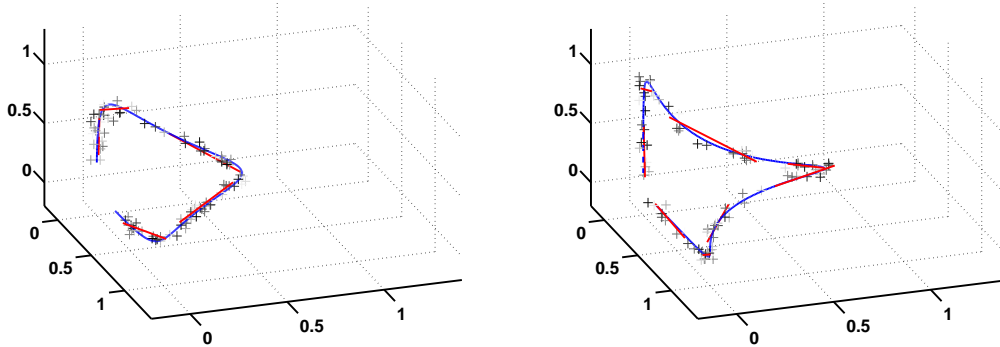


Figure 1: Approximation of MOUSSE at $t = 250$ (upper) and $t = 1150$ (lower) of a 100-dimensional submanifold. In this figure we project everything into three-dimensional space. The blue curve corresponds to true submanifold, the dots are noisy samples from the submanifold (the lighter dots are more dated than the darker dots), and the red line segments are the approximation with MOUSSE. As the curvature of the submanifold increases, MOUSSE also adapts in the number of line segments.

3 Multiscale Online Union of Subspace Estimation (MOUSSE)

In the following we describe the Multiscale Online Union of SubSpaces Estimation (MOUSSE) method, including the underlying multiscale model and online update approaches.

3.1 Multiscale union of subspaces model

MOUSSE uses a union of low-dimensional subsets $\widehat{\mathcal{M}}_t$ to approximate \mathcal{M}_t , and organizes these subsets using a tree structure. The idea for a multiscale tree structure is drawn from the multiscale harmonic analysis literature [36]. The leaves of the tree are subsets that are used for the submanifold approximation. Each node in the tree represents a local approximation to the submanifold at one scale. The parent nodes are subspaces that contain coarser approximations to the submanifold than their children. The subset associated with a parent node roughly covers the subsets associated with its two children.

More specifically, our approximation at each time t consists of a union of subspaces $\mathcal{S}_{j,k,t}$ that is organized using a tree structure. Here $j \in \{1, \dots, J_t\}$ denotes the scale or level of the subset in the tree, where J_t is the tree depth at time t , and $k \in \{1, \dots, 2^j\}$ denotes the index of the subset for that level. The approximation $\widehat{\mathcal{M}}_t$ at time t is given by:

$$\widehat{\mathcal{M}}_t = \bigcup_{(j,k) \in \mathcal{A}_t} \mathcal{S}_{j,k,t}. \quad (4)$$

where \mathcal{A}_t contains the indices of all leaf nodes used for approximation at time t . Also define \mathcal{T}_t to be the set of indices of all nodes in the tree at time t , with

$$\mathcal{A}_t \subset \mathcal{T}_t.$$

Each of these subsets lies on a low-dimensional hyperplane with dimension d and is parameterized as

$$\mathcal{S}_{j,k,t} = \{v \in \mathbb{R}^D : v = U_{j,k,t}z + c_{j,k,t}, \quad z^\top \Lambda_{j,k,t}^{-1}z \leq 1, \quad z \in \mathbb{R}^d\}. \quad (5)$$

where the notation \top denotes transpose of a matrix or vector. The matrix $U_{j,k,t} \in \mathbb{R}^{D \times d}$ is the subspace basis, and $c_{j,k,t} \in \mathbb{R}^D$ is the offset of the hyperplane from the origin. The diagonal matrix

$$\Lambda_{j,k,t} \triangleq \text{diag}\{\lambda_{j,k,t}^{(1)}, \dots, \lambda_{j,k,t}^{(d)}\} \in \mathbb{R}^{d \times d},$$

with $\lambda_{j,k,t}^{(1)} \geq \dots \geq \lambda_{j,k,t}^{(d)} \geq 0$, contains eigenvalues of the covariance matrix of the projected data onto each subspace. This parameter specifies the shape of the ellipsoid by capturing the spread of the data within the subset. In summary, the parameters for $\mathcal{S}_{j,k,t}$ are

$$\{U_{j,k,t}, c_{j,k,t}, \Lambda_{j,k,t}\}_{(j,k) \in \mathcal{T}_t},$$

and these parameters will be updated online.

In our tree structure, the leaf nodes of the tree also have two *virtual* children nodes that keep necessary information used when further partitioning is needed. The complexity of the approximation is defined to be the total number of subsets used for approximation at time t :

$$K_t \triangleq |\mathcal{A}_t|, \quad (6)$$

which is used as the complexity regularization term in (3)

$$\text{pen}(\widehat{\mathcal{M}}_t) \triangleq K_t. \quad (7)$$

The tree structure is illustrated in Figure 2.

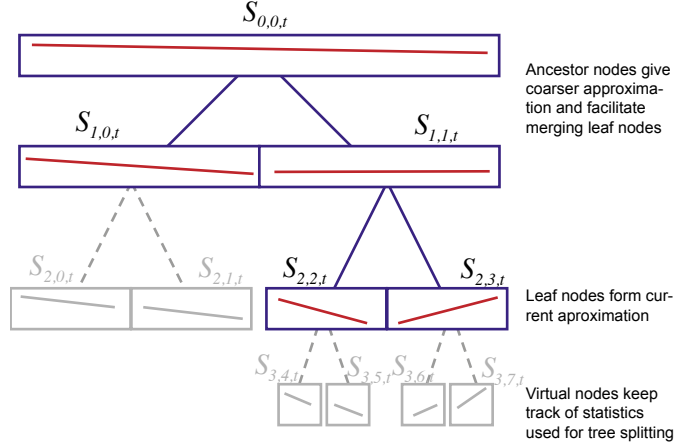


Figure 2: Illustration of tree structure for subspaces. The subspaces used in our approximation are $\{\mathcal{S}_{1,0,t} \cup \mathcal{S}_{2,2,t} \cup \mathcal{S}_{2,3,t}\}$.

3.2 MOUSSE Algorithm

When a new sample x_{t+1} becomes available, MOUSSE updates $\widehat{\mathcal{M}}_t$ to obtain $\widehat{\mathcal{M}}_{t+1}$. The update steps are presented in Algorithm 1; there are three main steps, detailed in the below subsections: (a) find the subset in the $\widehat{\mathcal{M}}_t$ which is closest to x_{t+1} , (b) update a tracking estimate of that closest subset and its ancestors, and (c) grow or prune the tree structure to preserve a balance between fit to data and complexity. We use $[z]_m$ to denote the m -th element of a vector z , and z^\top to denote the transpose of a matrix or vector z .

3.3 Distances for MOUSSE

To update the submanifold approximation, we first determine the affinity of x_{t+1} to each subset. We might use Euclidean distance, but this distance is problematic since in our approximation each subspace is local with boundary defined by an ellipsoid. Hence, a point can be close to a hyperplane but far away from the center of the ellipsoid. An alternative choice is the Mahalanobis distance, but this distance is only finite and well-defined for points lying in one of the low dimensional subspaces in our approximation. Since our construction is a piecewise linear approximation to a submanifold which may have some curvature, we anticipate many observations which are near but not in our collection of subspaces, and we need a well-defined, finite distance measure for such points.

To address these challenges, we introduce the *approximate Mahalanobis distance* of a point x to a subspace \mathcal{S} . Assume x with support Ω , parameter δ , and the parameters for a set \mathcal{S} is given by $\{U, c, \Lambda\}$. Define

$$U_\Omega \triangleq \mathcal{P}_\Omega U \in \mathbb{R}^{|\Omega| \times d}, \quad c_\Omega \triangleq \mathcal{P}_\Omega c \in \mathbb{R}^{|\Omega|},$$

and

$$x_\Omega = \mathcal{P}_\Omega x \in \mathbb{R}^{|\Omega|}.$$

Define the pseudoinverse operator that computes the coefficients of a vector in the subspace spanned by V as

$$V^\# \triangleq (V^\top V)^{-1} V^\top. \quad (8)$$

Algorithm 1 MOUSSE

- 1: Input:
error tolerance ϵ , step size α , relative weight μ
 - 2: Initialize tree structure, set $\epsilon_0 = 0$
 - 3: **for** $t = 0, 1, \dots$ **do**
 - 4: Given new data x_{t+1} and its support Ω_{t+1} , find the minimum distance set $\mathcal{S}_{j^*, k^*, t}$ according to $(j^*, k^*) = \arg \min_{(j, k) \in \mathcal{A}_t} \rho_{\delta_{j, k, t}}(x_{t+1}, \mathcal{S}_{j, k, t})$ using (13)
 - 5: Calculate: $e_{t+1} = \|\beta_{t, \perp}\|$ using (15)
 - 6: Update all ancestor nodes and closest virtual child node of (j^*, k^*) using Algorithm 2
 - 7: Calculate: $\epsilon_{t+1} = \alpha\epsilon_t + (1 - \alpha)e_{t+1}^2$
 - 8: Denote parent node of (j^*, k^*) as $(j^* - 1, k_p)$ and virtual child node closest to x_{t+1} as $(j^* + 1, k_v)$
 - 9: **if** $\epsilon_{t+1} > \epsilon$ and $d(x_{t+1}, \mathcal{S}_{j^*+1, k_v, t}) + \mu(K_t + 1) < e_{t+1}^2 + \mu K_t$ **then**
 - 10: Split (j^*, k^*) using Algorithm 3
 - 11: **end if**
 - 12: **if** $\epsilon_{t+1} < \epsilon$ and $d(x_{t+1}, \mathcal{S}_{j^*-1, k_p, t}) + \mu(K_t - 1) < e_{t+1}^2 + \mu K_t$ **then**
 - 13: Merge (j^*, k^*) and its sibling using Algorithm 4
 - 14: **end if**
 - 15: Update \mathcal{A}_t and \mathcal{T}_t
 - 16: **end for**
-

Algorithm 2 Update node

- 1: Input: node index (j, k) , α , δ and subspace parameters
 - 2: Calculate: β and β_{\perp} using (9) and (10)
 - 3: Update: $[c_{j, k, t+1}]_m = \alpha[c_{j, k, t}]_m + (1 - \alpha)[x_{t+1}]_m$, $m \in \Omega_{t+1}$
 - 4: Update: $\lambda_{j, k, t+1}^{(m)} = \alpha\lambda_{j, k, t}^{(m)} + (1 - \alpha)[\beta]_m^2$, $m = 1, \dots, d$
 - 5: Update: $\delta_{j, k, t+1} = \alpha\delta_{j, k, t} + (1 - \alpha)\|\beta_{\perp}\|^2 / (D - d)$
 - 6: Update basis $U_{j, k, t}$ using (modified) subspace tracking algorithm
-

Algorithm 3 Split node (j, k)

- 1: Turn two virtual children nodes $(j + 1, 2k)$ and $(j + 1, 2k + 1)$ of node (j, k) into leaf nodes
- 2: Initialize virtual nodes $(j + 1, 2k)$ and $(j + 1, 2k + 1)$:

$$\begin{aligned} k_1 &= 2k \\ k_2 &= 2k + 1 \\ c_{j+1, k_1, t+1} &= c_{j, k, t} + \sqrt{\lambda_{j, k, t}^{(1)}} u_{j, k, t}^{(1)} / 2 \\ c_{j+1, k_2, t+1} &= c_{j, k, t} - \sqrt{\lambda_{j, k, t}^{(1)}} u_{j, k, t}^{(1)} / 2 \\ U_{j+1, k_i, t+1} &= U_{j, k, t}, \quad i = 1, 2 \\ \lambda_{j+1, k_i, t+1}^{(1)} &= \lambda_{j, k, t}^{(1)} / 2, \quad i = 1, 2 \\ \lambda_{j+1, k_i, t+1}^{(m)} &= \lambda_{j, k, t}^{(m)}, \quad m = 2, \dots, d, \quad i = 1, 2 \end{aligned}$$

Algorithm 4 Merge (j, k) and its sibling

- 1: Make the parent node of (j, k) into a leaf node
 - 2: Make (j, k) and its sibling into virtual children nodes of the newly created leaf
 - 3: Delete all four virtual children nodes of (j, k) and its sibling
-

When U is an orthogonal matrix, we have $U^{\#} \equiv U^{\top}$, but in general $U_{\Omega}^{\#} \neq U_{\Omega}^{\top}$. Let

$$\beta = U_{\Omega}^{\#}(x_{\Omega} - c_{\Omega}), \quad (9)$$

and

$$\beta_{\perp} = (I - U_{\Omega}U_{\Omega}^{\#})(x_{\Omega} - c_{\Omega}). \quad (10)$$

In this definition, β is the projection of x on U , and $\|\beta_{\perp}\|$ captures the energy of the projection residual. We denote Euclidean distance between x with support Ω and the subspace where \mathcal{S} lies on as

$$d(x, \mathcal{S}) \triangleq \|x_{\Omega} - c_{\Omega} - U_{\Omega}U_{\Omega}^{\#}(x_{\Omega} - c_{\Omega})\|^2 = \|\beta_{\perp}\|^2. \quad (11)$$

Next we introduce the *approximate Mahalanobis distance*, which is a hybrid of Euclidean distance and Mahalanobis distance. Mahalanobis distance is commonly used for data classification, which measures the quadratic distance of x to a set \mathcal{S} of data with mean $c = \mathbb{E}\{x\}$ and covariance $\Sigma = \mathbb{E}\{(x - c)(x - c)^{\top}\}$. Specifically, the Mahalanobis distance is defined as

$$\varrho(x, \mathcal{S}) = (x - c)^{\top}\Sigma^{-1}(x - c).$$

Assuming the covariance matrix has a low-rank structure with d large eigenvalues and $D - d$ small eigenvalues, we can write the eigendecomposition of the covariance matrix Σ as

$$\Sigma \triangleq \begin{bmatrix} U & U_{\perp} \end{bmatrix} \Lambda \begin{bmatrix} U & U_{\perp} \end{bmatrix}^{\top} = U\Lambda_1U^{\top} + U_{\perp}\Lambda_2U_{\perp}^{\top},$$

where $\Lambda = \text{diag}\{\lambda_1, \dots, \lambda_D\}$, $\lambda_1 \geq \dots \geq \lambda_D$, $\Lambda_1 = \text{diag}\{\lambda_1, \dots, \lambda_d\}$, $\Lambda_2 = \text{diag}\{\lambda_{d+1}, \dots, \lambda_D\}$. If we further assume that the $D - d$ small eigenvalues are all approximately equal to δ , *i.e.*, $\Lambda_2 \approx \delta I$, then

$$\varrho(x, \mathcal{S}) \approx (x - c)^{\top}U\Lambda_1^{-1}U^{\top}(x - c) + \delta^{-1}\|U_{\perp}^{\top}(x - c)\|^2. \quad (12)$$

From here we introduce the approximate Mahalanobis distance when the data is low-dimensional:

$$\rho_{\delta}(x, \mathcal{S}) \triangleq \beta^{\top}\Lambda^{-1}\beta + \delta^{-1}\|\beta_{\perp}\|^2. \quad (13)$$

Note that when the data is complete, $\rho_{\delta}(x, \mathcal{S})$ is equal to the right-hand-side of (12). With missing data, $\rho_{\delta}(x, \mathcal{S})$ is an approximation to $\varrho(x, \mathcal{S})$.

3.4 Update subset parameters

When updating subspaces, we can update all subspaces in our multiscale representation and make the update step-size to be inversely proportional to the approximate Mahalanobis distance between the new sample and each subspace, which we refer to as the “update-all” approach. Alternatively, we can just update the subspace closest to x_{t+1} , its virtual children, and all its ancestor nodes, which we refer to as the “update-nearest” approach. The update-all approach is computationally more expensive, especially for high dimensional problems, so we focus our attention on the greedy update-nearest approach. The below approaches extend readily to the update-all setting, however.

With the approximate Mahalanobis distance defined above, we can find the subset with minimum distance to the new datum x_{t+1} :

$$(j^*, k^*) = \arg \min_{(j,k)} \rho_{\delta_{j,k,t}}(x_{t+1}, \mathcal{S}_{j,k,t}),$$

and then update the parameters of that subset, all its ancestors in the tree, and its two virtual children. The update algorithm is summarized in Algorithm 2 which denotes the parameters associated with $\mathcal{S}_{j^*,k^*,t}$ as (c, U, Λ, δ) , and drops the j^* , k^* , and t indices for simplicity of presentation. The update of the center c , Λ and δ are straightforward.

Using the definition in (2), we have

$$\mathcal{P}_{\widehat{\mathcal{M}}_t} x = \mathcal{P}_{\Omega}^{\top}U_{\Omega}U_{\Omega}^{\#}(x_{\Omega} - c_{\Omega}) + c; \quad (14)$$

that is, the projection onto the submanifold approximation is the projection onto the nearest subset. We further define the *instantaneous approximation error* of the submanifold at time t as:

$$e_t \triangleq \|\mathcal{P}_{\Omega_t}(x_t - \mathcal{P}_{\widehat{\mathcal{M}}_t}x_t)\|, \quad (15)$$

and note that this is equivalent to the norm of the orthogonal projection of x_t onto $\mathcal{S}_{j^*,k^*,t}$, denoted $\beta_{t,\perp}$; *i.e.*,

$$e_t \equiv \|\beta_{t,\perp}\|. \quad (16)$$

Next we will focus on three approaches for updating U .

3.4.1 GROUSE

To use GROUSE subspace tracking in this context, we approximate the first term in (3) as

$$F(\mathcal{M}) = \sum_{i=1}^t \alpha^{t+1-i} \|\mathcal{P}_{\Omega_i}(x_i - \mathcal{P}_{\widehat{\mathcal{M}}_i} x_i)\|^2 + \|\mathcal{P}_{\Omega_{t+1}}(x_{t+1} - \mathcal{P}_{\mathcal{M}} x_{t+1})\|^2. \quad (17)$$

Note the first term is a constant with respect to \mathcal{M} , so we need only to consider the second term in computing an update. The basic idea is now to take a step in the direction of the instantaneous gradient of this cost function. Since \mathcal{M} is constrained to be a union of subsets and the projection operator maps to the closest subset, this task corresponds to the basis update of GROUSE [15] with the cost function

$$f(U) \triangleq \min_a \|\mathcal{P}_{\Omega_{t+1}}(x_{t+1} - Ua - c)\|^2 \quad (18)$$

(assuming U is orthonormal and including the offset vector c). Following the same derivation as in [15], we have that

$$\frac{df}{dU} = -2\mathcal{P}_{\Omega_{t+1}}(x_{t+1} - c - U\beta)\beta^\top \triangleq -2r\beta^\top, \quad (19)$$

where β is defined in (9), and

$$r = \mathcal{P}_{\Omega_{t+1}}(x_{t+1} - c - U\beta).$$

The gradient on the Grassmannian is given by

$$\nabla f = (I - UU^\top) \frac{df}{dU} = -2(I - UU^\top)r\beta^\top = -2r\beta^\top,$$

since $U^\top r = 0$. We obtain that the update of U_t using the Grassmannian gradient is given by

$$U_{t+1} = U_t + \frac{\cos(\xi\eta) - 1}{\|\beta\|^2} U_t \beta \beta^\top + \sin(\xi\eta) \frac{r}{\|r\|} \frac{\beta^\top}{\|\beta\|}.$$

where $\eta > 0$ is the step-size, and $\xi = \|r\| \|U_t \beta\|$. The step-size η is chosen to be $\eta = \eta_0 / \|x_{t+1}\|$, for a constant $\eta_0 > 0$.

3.4.2 PETRELS

Let $x_{(1)}, x_{(2)}, x_{(3)}, \dots$ denote a subsequence of the data such that each $x_{(i)}$ was drawn from the (j^*, k^*) node in our multiscale approximation, and let n_t denote the length of this subsequence. Then we can approximate $F(\mathcal{M})$ as

$$F(\mathcal{M}) = \sum_{i=1}^{n_t} \alpha^{n_t-i} \min_z \|\mathcal{P}_{\Omega_{(i)}}(x_{(i)} - c - Uz)\|^2, \quad (20)$$

where, as before, U and c correspond to the subspace parameters for subset $\mathcal{S}_{j^*, k^*, t}$. The minimization of $F(\mathcal{M})$ in (20) can be accomplished using the PETRELS algorithm [37], yielding a solution which can be expressed recursively as follows. Denoting by $[U]_m$ the m -th column of U , we have the update of U given by

$$[U_{t+1}]_m = [U_t]_m + I_{m \in \Omega_t} ([U_t a_{t+1}]_m - a_{t+1}^\top [U_t]_m) (R_{m,t+1})^\# a_{t+1}, \quad (21)$$

for $m = 1, \dots, D$, where I_A is the indicator function for event A , $(\cdot)^\#$ denotes pseudo-inverse of a matrix, and

$$a_{t+1} = (U_t^\top \mathcal{P}_{\Omega_{t+1}} U_t)^\# U_t^\top x_{t+1}.$$

The second-order information in $R_{m,t+1}$ can be computed recursively as

$$(R_{m,t+1})^\# = \alpha^{-1} (R_{m,t})^\# + \frac{\alpha^{-2} p_{m,t+1}}{1 + \alpha^{-1} a_{t+1}^\top (R_{m,t})^\# a_{t+1}} (R_{m,t})^\# a_t a_t^\top (R_{m,t})^\#. \quad (22)$$

Note that PETRELS does not guarantee the orthogonality of U_{t+1} , which is important for quickly computing projections onto our submanifold approximation. To obtain orthonormal U_{t+1} , we may apply Gram-Schmidt orthonormalization after each update. We refer to this modification of PETRELS as *PETRELS-GS*. This orthogonalization requires an extra computational cost on the order of $\mathcal{O}(Dd^2)$ and may compromise the continuity of U_t , i.e., the Frobenius norm $\|U_{t+1} - U_t\|_F$ after the orthogonalization may not be small even when the corresponding subspaces are very close [38]. As a result, this orthogonalization may change the optimality of U_t . A faster orthonormalization (FO) strategy with less computation which also preserves the continuity of U_t is given in [38]. We refer to this FO strategy combined with PETRELS as *PETRELS-FO*.

3.4.3 Computational complexity

For each update with complete data (the maximum computational complexity), the computational complexity of GROUSE is on the order of $\mathcal{O}(Dd)$, PETRELS-GS is $\mathcal{O}(Dd^2)$, and PETRELS-FO is $\mathcal{O}(Dd)$. More details about the relative performance of these three subspace update methods can be found in Section 6.

3.5 Tree structure update

When the curvature of the submanifold changes and cannot be sufficiently characterized by the current subset approximations, we must perform adaptive model selection. This can be accomplished within our framework by updating the tree structure – growing the tree or pruning the tree, which we refer to as “splitting” and “merging” branches, respectively. Previous work has derived finite sample bounds and convergence rates of adaptive model selection in nonparametric time series prediction [39].

To decide whether to change the tree structure, we introduce the *average approximation error*:

$$\epsilon_t \triangleq \sum_{i=1}^t \alpha^{t-i} \|\mathcal{P}_{\Omega_i}(x_i - \mathcal{P}_{\widehat{\mathcal{M}}_i} x_i)\|^2 = \alpha \epsilon_{t-1} + (1 - \alpha) e_t^2. \quad (23)$$

This error is an approximation to the first term in (3), where we replace $\mathcal{P}_{\widehat{\mathcal{M}}}$ with the projection onto a sequence of approximations $\mathcal{P}_{\widehat{\mathcal{M}}_i}$. We will consider changing the tree structure when ϵ_t is greater than our prescribed error tolerance $\epsilon > 0$.

Splitting tree branches increases the resolution of the approximation at the cost of higher estimator complexity. Merging reduces resolution but lowers complexity. When making decisions on splitting or merging, we take into consideration the approximation errors as well as the model complexity (the number of subspaces K_t used in the approximation). This is related to complexity-regularized tree estimation methods [36, 40, 41] and the notion of minimum description length (MDL) in compression theory [42, 43]. In particular, we use the sum of the average fitting error and a penalty proportional to the number of subspaces used for approximation as the cost function when deciding to split or merge. The splitting and merging are summarized in Algorithm 3 and Algorithm 4. The splitting process mimics the k -means algorithm. In these algorithms, note that for node (j, k) the parent is node $(j - 1, \lfloor k/2 \rfloor)$ and the sibling node is $(j, k + 1)$ for k even or $(j, k - 1)$ for k odd.

3.6 Initialization

To initialize MOUSSE, we assume a small initial training set of samples, and perform a nested bi-partition of the training data set to form a tree structure, as shown in Figure 2. The root of the tree represents the entire data set, and the children of each node represent a bipartition of the data in the parent node. The bipartition of the data can be performed by the k -means algorithm. We start with the entire data, estimate the sample covariance matrix, perform an eigendecomposition, extract the d -largest eigenvectors and eigenvalues and use them for $U_{1,1,0}$ and $\Lambda_{1,1,0}$, respectively. The average of the $(D - d)$ minor eigenvalues are used for $\delta_{1,1,0}$. If the approximation error is greater than the prescribed error tolerance ϵ , we further partition the data into two clusters using k -means (for $k = 2$) and repeat the above process. We keep partitioning the data until $\delta_{j,k,0}$ is less than ϵ for all leaf nodes. Then we further partition the data one level down and form the virtual nodes. This tree construction is similar to that used in [14].

In principle, it is possible to bypass this training phase and just initialize the tree with a single root node and two random virtual children nodes. However, the training phase makes it much easier to select algorithm parameters such as ϵ and provides more meaningful initial virtual nodes, thereby shortening the “burn in” time of the algorithm.

3.7 Choice of parameters

In general, α should be close to 1, as in the Recursive Least Squares (RLS) algorithm [44]. In the case when the submanifold changes quickly, we would expect smaller weights for approximation based on historical data and thus a smaller α . In contrast, a slowly evolving submanifold requires a larger α . In our experiments, α ranges from 0.8 to 0.95. ϵ controls the data fit error, which varies from problem to problem according to the smoothness of the submanifold underlying the data and the noise variance. Since the tree’s complexity is controlled and $\text{pen}(\mathcal{M})$ in (3) is roughly on the order of $\mathcal{O}(1)$, we usually set μ close to ϵ .

4 Changepoint detection

We are interested in detecting changes to the submanifold that arise abruptly and change the statistics of the data. When the submanifold varies slowly in time, MOUSSE described in Section 3 can track the submanifold and produce a sequence of stationary tracking errors. When an abrupt change occurs, MOUSSE loses track of the manifold and results in an abrupt increase in tracking errors. Hence, using tracking errors, we can develop a changepoint detection algorithm to detect abrupt changes in the submanifold.

4.1 CUSUM procedure

We adopt the widely used statistical CUSUM procedure [2, 45] for changepoint detection. In particular, we assume that ν_0 is a normal distribution with mean μ_0 and variance σ_0^2 , and ν_1 is a normal distribution with mean μ_1 and the same variance σ_0^2 . Then we can formulate the changepoint detection problem as the following hypothesis test:

$$\begin{aligned} \text{H}_0 : e_1, \dots, e_t &\sim \mathcal{N}(\mu_0, \sigma_0^2) \\ \text{H}_1 : e_1, \dots, e_\kappa &\sim \mathcal{N}(\mu_0, \sigma_0^2), \quad e_{\kappa+1}, \dots, e_t \sim \mathcal{N}(\mu_1, \sigma_0^2) \end{aligned} \quad (24)$$

We assume μ_0 and σ_0^2 are known since typically there is enough normal data to estimate these parameters. (When the training phase is too short for this to be the case, these quantities can be estimated online, as described in [46].) However, we assume μ_1 is unknown since the magnitude of the changepoint can vary from one instance to another. In forming the detection statistic, we replace μ_1 by its maximum likelihood estimate (for each fixed changepoint time $\kappa = k$):

$$\hat{\mu}_1 = \frac{S_t - S_k}{t - k},$$

where

$$S_t \triangleq \sum_{i=1}^t e_i.$$

This leads to the generalized CUSUM procedure, which computes the CUSUM statistic at each time t and stops the first time when the statistic hits threshold b :

$$T = \inf \left\{ t \geq 1 : \max_{t-w \leq k < t} \frac{|(S_t - S_k) - \mu_0(t - k)|}{\sigma_0 \sqrt{t - k}} \geq b \right\}, \quad (25)$$

where w is a time-window length such that we only consider the most recent w errors for changepoint detection, and the threshold b is chosen to control the false-alarm-rate, which is characterized using average-run-length (ARL) in the changepoint detection literature [47]. Typically we would choose w to be several times (for example, 5 to 10 times) of the expected detection, then the window length will almost have no effect on the detection delay [48]. This threshold choice is detailed in Section 5.5.

4.2 Distribution of e_t

In deriving the CUSUM statistics we have assumed that e_t are i.i.d. Gaussian distributed. A fair question to ask is whether e_t is truly Gaussian distributed, or even to ask whether e_t is a good statistic to use. Consider the complete data case.

$$\hat{e}_t \triangleq \mathcal{P}_{\widehat{\mathcal{M}}_t}(x_t) - x_t. \quad (26)$$

In general \hat{e}_t has non-zero mean, since we approximate the manifold using a union of subspaces. Moreover, due to curvature of the manifold, in general each dimension of the error vector, $[\hat{e}_t]_m$, are not independent and the variances of $[\hat{e}_t]_m$ may not be identical. In fact, no closed form expression for the distribution of $e_t = \|\hat{e}_t\|$ exists.

However, a Gaussian distribution is a good approximation for the distribution of e_t , since when D is large, e_t averages over errors across D dimensions. The QQ-plot of e_t from one of our numerical examples in Section 6 when $D = 100$ is shown in Figure 3. We will also demonstrate in Section 6.4 that the theoretical approximation for ARL using Gaussian assumption for e_t is quite accurate.

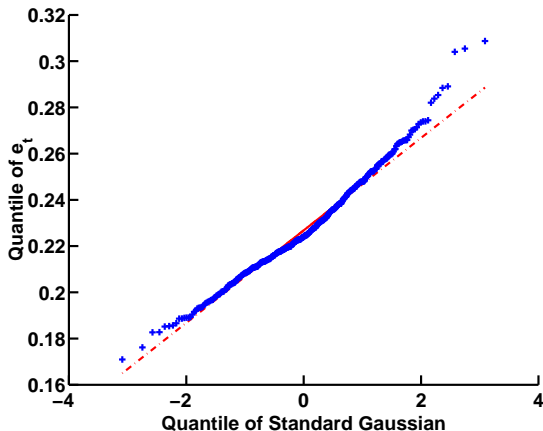


Figure 3: Q-Q plot of e_t , for a $D = 100$ submanifold.

5 Performance Analysis

In this section, we first study the performance of MOUSSE, and then study the choice for the threshold parameter of the changepoint detection algorithm and provide theoretical approximations. A complete proof of convergence of MOUSSE (or GROUSE or PETRELS) is hard since the space of submanifold approximations we consider is non-convex. Nevertheless, we can still characterize several aspects of our approach.

In Sections 5.1 and 5.2 below, we assume that there is complete data, and we restrict our approximation to a single subspace so that $K_t = 1$. Assume the mean and covariance matrix of the data are given by c^* and Σ^* , respectively. Assume the covariance matrix has low-rank structure: $\Sigma^* = \text{diag}\{\lambda_1^*, \dots, \lambda_D^*\}$ with $\lambda_m = \delta^*$ for $m = d + 1, \dots, D$.

5.1 Optimality of estimator for c^*

When there is only one subspace and the data are complete, the cost function (3) without the penalty term becomes

$$\min_{U, c} \sum_{i=1}^t \alpha^{t-i} \|(I - UU^\top)(x_i - c)\|^2. \quad (27)$$

By writing each term as $\|(I - UU^\top)(x_i - c)\|^2 = \|(I - UU^\top)(x_i - \bar{x} + \bar{x} - c)\|^2$, we can write the cost function as

$$\text{tr}[(I - UU^\top)S(I - UU^\top)] + \frac{1 - \alpha^t}{1 - \alpha} (\bar{x} - c)^\top (I - UU^\top) (\bar{x} - c), \quad (28)$$

where

$$\bar{x} = \sum_{i=1}^t \alpha^{t-i} x_i, \quad S = \sum_{i=1}^t \alpha^{t-i} (x_i - \bar{x})(x_i - \bar{x})^\top. \quad (29)$$

Since the second term in (28) is quadratic in c , it is minimized by $c = \bar{x}$. Also note that in this case the optimization problem (3) decouples in U and c . Since our online update for c_t is given by $c_{t+1} = \alpha c_t + (1 - \alpha)x_t$, we have

$$c_t = (1 - \alpha) \sum_{i=1}^t \alpha^{t-i} x_i + \alpha^t c_0,$$

where the term $\alpha^t c_0$ is a bias introduced by initial condition c_0 . Since $\mathbb{E}\{x_t\} = c^*$, $\mathbb{E}\{c_t\} \rightarrow (1 - \alpha) \cdot \frac{1}{1 - \alpha} c^* = c^*$. Hence our estimator for c is proportional to the minimizer for (3) and is asymptotically unbiased.

5.2 Consistency of estimators for Λ^* and δ^*

In the following, we show that when there is complete data, if we have correct $U = U^*$, then for each sample x_t , its projection $[\beta_t]_m$ is an unbiased estimator for λ_m^* , and $\|\beta_{t,\perp}\|^2$ is an unbiased estimator for $\sum_{m=d+1}^D \lambda_m^*$. First note

$$\begin{aligned} \mathbb{E}\{|[\beta_t]_m|^2\} &= \mathbb{E}\{[e_m^\top U^\top (x_t - c)]^2\} = e_m^\top U^\top \Sigma^* U e_m \\ &= \lambda_m^* [U]_m^\top [U]_m = \lambda_m^*, \end{aligned} \quad (30)$$

for $m = 1, \dots, d$, where \mathbf{e}_m denotes the m -th row of an identity matrix. We also have that

$$\begin{aligned}\mathbb{E}\{\|\beta_{t,\perp}\|^2\} &= \mathbb{E}\{\|(I - UU^\top)(x_t - c)\|^2\} \\ &= \text{tr}\{(I - UU^\top)\Sigma^*(I - UU^\top)\} \\ &= \sum_{m=d+1}^D \lambda_m^*.\end{aligned}\tag{31}$$

Then from the MOUSSE update equations, as $t \rightarrow \infty$

$$\mathbb{E}\{\lambda_t^{(m)}\} = \mathbb{E}\{(1 - \alpha) \sum_{i=1}^t \alpha^{t-i} \|\beta_t\|_m^2 + \alpha^t \lambda_0^{(m)}\} \rightarrow \lambda_m^*,\tag{32}$$

for $m = 1, \dots, d$ and

$$\mathbb{E}\{\delta_t\} = \mathbb{E}\{(1 - \alpha) \sum_{i=1}^t \alpha^{t-i} \|\beta_{t,\perp}\|^2 / (D - d) + \alpha^t \delta_0\} \rightarrow \frac{1}{D-d} \sum_{m=d+1}^D \lambda_m^* = \delta^*.\tag{33}$$

Hence our estimators for λ_m^* and δ^* are asymptotically unbiased. Moreover, from (16), we have that

$$\mathbb{E}\{e_t^2\} = \mathbb{E}\{\|\beta_{t,\perp}\|^2\} = \sum_{m=d+1}^D \lambda_m^*.\tag{34}$$

Hence, the average approximation error of MOUSSE also converges

$$\mathbb{E}\{\epsilon_t\} = (1 - \alpha) \sum_{i=1}^t \alpha^{t-i} \mathbb{E}\{e_i^2\} \rightarrow \sum_{m=d+1}^D \lambda_m^*.$$

5.3 MOUSSE approximation errors

As mentioned earlier, our multiscale subset model is closely related to geometric multiresolution analysis (GMRA) [14]. In that work, the authors characterize the favorable approximation capabilities of the proposed multiscale model. In particular, they prove that the magnitudes of the geometric wavelet coefficients associated with their algorithm decay asymptotically as a function of scale, so a collection of data lying on a smooth submanifold can be well-approximated with a small number (depending on the submanifold curvature) of relatively large geometric wavelets. These geometric wavelets are akin to the leaf nodes in our approximation, so the approximation results of [14] suggest that our model admits accurate approximations of data on smooth submanifolds with a small number of leafs.

5.4 Missing data

In Section 5.2 we have shown that with full data, our estimators are consistent. Note these estimators depend on β and β_\perp . In the following we show that β and β_\perp , when using a missing data projection, are close to their counterparts when using a complete data projection. Hence, when the fraction of missing data is not large, the performance of MOUSSE with missing data is also consistent. In this section, we omit the subscripts j , k and t , and denote Ω_t by Ω to simplify notation. Define the coherence of the basis U as [49]

$$\text{coh}(U) = \frac{D}{d} \max_m \|U \mathcal{P}_U \mathbf{e}_m\|_2^2.\tag{35}$$

Theorem 1. *Let $\varepsilon > 0$. Given $x = v + w$, and w is a white Gaussian noise with zero mean and covariance matrix $\sigma^2 I_{D \times D}$. Let $\beta = U^\top(x - c)$, and $\beta_\Omega = U_\Omega^\#(x_\Omega - c_\Omega)$. If for some constant $\ell \in (0, 1)$,*

$$|\Omega| \geq \max \left\{ \frac{8}{3} \text{coh}(U) d \log(2d/\varepsilon), \frac{4}{3} \frac{D}{(1 - \ell) \log(2D/\varepsilon)} \right\},$$

then with probability at least $1 - 3\varepsilon$,

$$\|\beta_\Omega - \beta\|_2^2 \leq 2 \frac{(1 + \theta)^2}{(1 - \ell)^2} \cdot \frac{d}{|\Omega|} \cdot \text{coh}(U) \|q\|^2 + \sigma^2 \frac{(64/9)D^2}{(1 - \ell)^2 |\Omega|^2},\tag{36}$$

where

$$\theta = \sqrt{2 \frac{D \max_{n=1}^D |[q]_n|^2}{\|q\|^2} \log(1/\varepsilon)},$$

and $q = (I - UU^\top)(v - c)$.

This theorem shows that the fraction of non-zero entries, $|\Omega|$, should be on the order of $\max\{d \log d\}$ or $\{D/\log(D)\}$ for accurate estimation of β_Ω . We have a similar bound for the difference between β_\perp and $\beta_{\Omega,\perp}$, based on similar techniques used to derive Theorem 1. The first term in the error bound (36) is proportional to $\|q\|$, the approximation error related to the distance of v from U , and the second term in (36) is due to noise.

Proof of this theorem can be found in Appendix A. This theorem can be viewed as an extension of [49, Theorem 1] to include noise, and also to directly bound $\|\beta - \beta_\Omega\|$ instead of bounding $\|v_\Omega - U_\Omega \beta_\Omega\|$ using $\|v - U\beta\|$.

5.5 Choice of threshold for changepoint detection

In accordance with standard changepoint detection notation, denote by \mathbb{E}^∞ the expectation when there is no change, i.e., \mathbb{E}_{H_0} , and by \mathbb{E}^k the expectation when there is a changepoint at $\kappa = k$, i.e., $\mathbb{E}_{H_1, \kappa=k}$. The performance metric for a changepoint detection algorithm is typically characterized by the *expected detection delay* $\sup_{k \geq 0} \mathbb{E}^k\{T - k | T > k\}$ and the *average-run-length* (ARL) $\mathbb{E}^\infty\{T\}$ [47]. Typically we use $\mathbb{E}^0\{T\}$ as a performance metric since it is an upper bound for $\sup_{k \geq 0} \mathbb{E}^k\{T - k | T > k\}$. Note that the CUSUM statistic (25) is equivalent to

$$T = \inf\{t \geq 1 : \max_{t-w \leq k < t} \frac{|\tilde{S}_t - \tilde{S}_k|}{\sqrt{t-k}} \geq b, \} \quad (37)$$

where $\tilde{S}_t = \sum_{i=1}^t (e_i - \mu_0)/\sigma_0$. Under H_0 , we have $(e_i - \mu_0)/\sigma_0$ i.i.d. Gaussian distributed with zero mean and unit variance. Using the results in [50], we have the following approximation. When $b \rightarrow \infty$,

$$\mathbb{E}^\infty\{T\} \sim \frac{(2\pi)^{1/2} \exp\{b^2/2\}}{b \int_0^b x \nu^2(x) dx}, \quad (38)$$

where $\nu(x) = \frac{(2/x)[\Phi(x/2)-0.5]}{(x/2)\Phi(x/2)+\phi(x)/2}$ [48], $\phi(x)$ and $\Phi(x)$ are the pdf and cdf of the normal random variable with zero mean and unit variance. We will demonstrate in Section 6.4 that this asymptotic approximation is fairly accurate even for finite b , which allows us to choose the changepoint detection threshold to achieve a target ARL without parameter tuning.

6 Numerical Examples

In this section, we present several numerical examples, first based on simulated data, and then real data, to demonstrate the performance of MOUSSE in tracking a submanifold and detecting changepoints. We also verify that our theoretical approximation to ARL in Section 5.5 is quite accurate.

6.1 Tracking a static submanifold

We first study the performance of MOUSSE tracking a static submanifold. The dimension of the submanifold is $D = 100$ and the intrinsic dimension is $d = 1$. Fixing $\theta \in [-2, 2]$, we define $v(\theta) \in R^D$ with its n -th element

$$[v(\theta)]_n = 1/\sqrt{2\pi} e^{-(z_n - \theta)^2/(2\gamma^2)}, \quad (39)$$

where $\gamma = 0.6$, and $z_n = -2 + 4n/D$, $n = 1, \dots, 100$, corresponds to regularly spaced points between -2 and 2 . This static submanifold is sampled by sampling different $\theta \in [-2, 2]$ and generating corresponding points on the submanifold according to (39). The observation x_t is obtained from (1), where the noise variance is $\sigma^2 = 4 \times 10^{-4}$. We set parameter values as $\alpha = 0.95$, $\epsilon = 0.1$, $\mu = 0.03$, and use PETRELS-FO. Figure 4 demonstrates that MOUSSE is able to track a static submanifold and reach the steady state quickly from a coarse initialization. In Figure 4 and the following numerical examples, the expected instantaneous fitting error is evaluated using $N = 1200$ draws from \mathcal{M} , denoted y_1, \dots, y_N and computing the Monte Carlo estimate

$$\mathbb{E}\{e_t\} \approx \frac{1}{N} \sum_{i=1}^N d(y_i, \mathcal{S}_i), \quad (40)$$

where \mathcal{S}_i denotes the minimum distance subset to y_i .

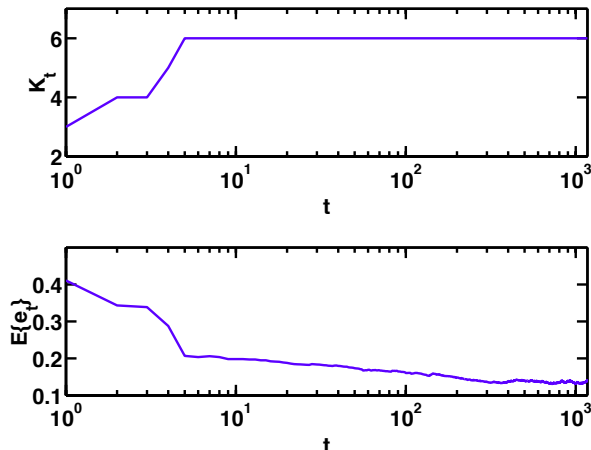


Figure 4: MOUSSE tracking a static submanifold with $D = 100$ and $d = 1$.

6.2 Tracking a slowly time varying submanifold

Next we track a slowly time varying submanifold using MOUSSE, in a similar setting to Section 6.1 where $D = 100$ and $d = 1$. The submanifold is also generated using (39), except that now we let γ to be time varying:

$$\gamma_t = \begin{cases} 0.6 - \gamma_0 t & t = 1, 2, \dots, s \\ 0.6 - \gamma_0(2s - t) & t = s + 1, s + 2, \dots, 2s \end{cases} \quad (41)$$

where parameter γ_0 controls how fast the submanifold changes. We choose $\gamma_0 = 2 \times 10^{-4}$, $s = 1000$ with 20% missing data, and the parameters for MOUSSE are $\mu = 0.03$, $\epsilon = 0.1$, and $\alpha = 0.9$. The result of the tracking can be found in an illustrative video in <http://nislabs.ee.duke.edu/MOUSSE/index.html>. Snapshots of this video at time $t = 250$ and $t = 1150$ are shown in Figure 1. In this display, the dashed line corresponds to the true submanifold, the red lines correspond to the estimated union of subspaces, and the + signs correspond to the past 500 samples, with darker colors corresponding to more recent observations. From this video, it is clear that we are effectively tracking the dynamics of the submanifold, and keeping the representation parsimonious so the number of subspaces used by our model is proportional to the curvature of the submanifold, and as the curvature increases and decreases, the number of subspaces used in our approximation similarly increases and decreases. The number of subspaces K_t and fitting error as a function of time are shown in Figure 5. The red line in Figure 5 corresponds to ϵ . Note that MOUSSE is able to track the submanifold, in that it can maintain a stable number of leaf nodes in the approximation and meet the target error tolerance ϵ .

6.3 Comparison of tracking algorithms

We also compare the performance of different tracking algorithms presented in Section 3.4: GROUSE, PETRELS-GS and PETRELS-FO. We use $\mathbb{E}\{e_t\}$ defined in (40) as a comparison metric. In comparing the three methods, we set the parameters for each tracking algorithm such that the algorithm has the best possible performance. The comparison is displayed in Figure 6, where the horizontal axis is the submanifold changing rate γ , the vertical axis is the percentage of missing data, and the brightness of each block corresponds to $\mathbb{E}\{e_t\}$. In Figure 6, PETRELS-FO performs better than GROUSE and PETRELS-GS, especially with a large percentage of missing data. Also note that for PETRELS-FO, the optimal parameters are fairly stable for various combinations of submanifold changing rate and percentage of missing data: the optimal parameters are $\alpha \approx 0.9$, $\mu \approx 0.2$, and $\epsilon \approx 0.1$. Considering its lower computational cost and ease of parameter tuning, we use PETRELS-FO with MOUSSE for the remaining experiments in this paper.

6.4 Changepoint detection example

To verify our theoretical approximation for ARL, we perform Monte Carlo simulation. Direct simulation of T to obtain $\mathbb{E}^\infty\{T\}$ is very time-consuming because we typically want to choose a b such that $\mathbb{E}^\infty\{T\}$ is on the order of 10000. Hence we use an indirect simulation method commonly used in changepoint detection [48]. The indirect

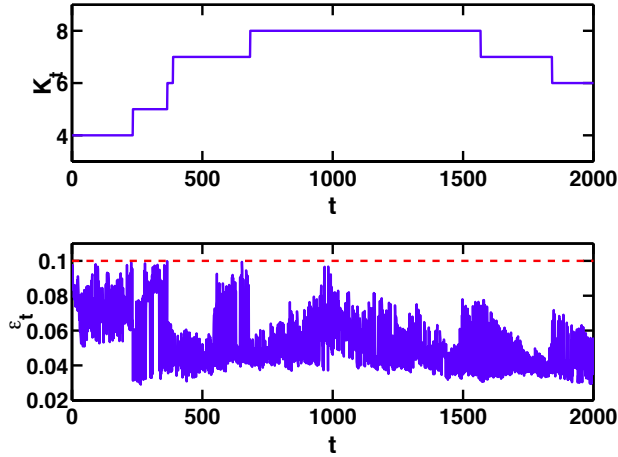


Figure 5: MOUSSE tracking a slowly evolving submanifold with $D = 100$ and $d = 1$. Dashed red line depicts target error tolerance ϵ .

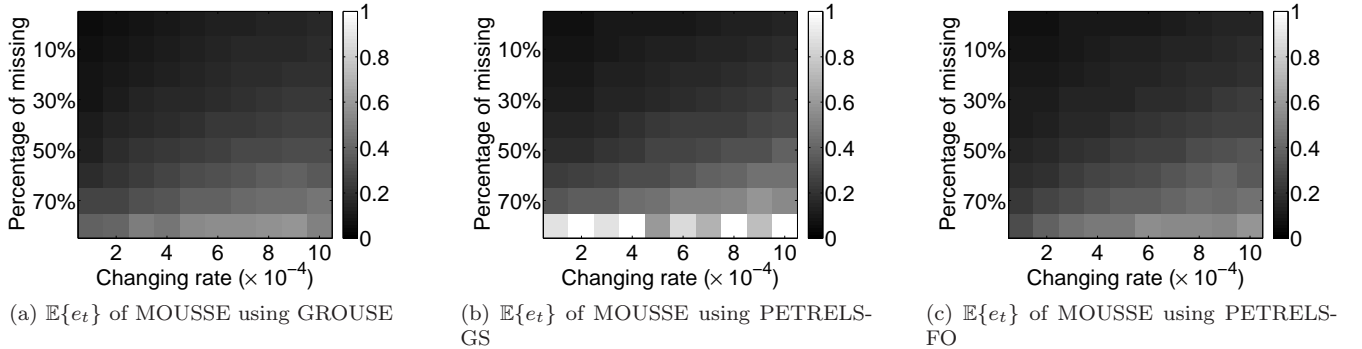


Figure 6: MOUSSE tracking a slowly varying submanifold using: (a) GROUSE, (b) PETRELS-GS and (c) PETRELS-FO. Horizontal axis corresponds to rate of submanifold's change and vertical axis corresponds to fraction of data missing. Brightness corresponds to $\mathbb{E}\{e_t\}$.

method is based on the fact that when there is no changepoint, for large b , T is typically exponentially distributed. Hence, we have under H_0 ,

$$\mathbb{P}\{T > m\} = e^{-m/E^\infty\{T\}},$$

which means we can simulate $\mathbb{P}\{T > m\}$ for fixed m and b , and then obtain under H_0

$$\mathbb{E}^\infty\{T\} = -m/\log\mathbb{P}\{T > m\}. \quad (42)$$

Using this formula, we generate 10000 Monte Carlo (MC) trials, each a slowly time-varying submanifold of duration $t = 500$. Then we use MOUSSE to track the submanifold, and obtain a sequence of errors. We form the CUSUM statistics using this sequence of errors, and find the fraction of sequences such that

$$\max_{1 \leq t \leq 500} \max_{t-w \leq k \leq t} \frac{1}{\sigma_0} \frac{|(S_t - S_k) - \mu_0(t - k)|}{\sqrt{t - k}} \geq b,$$

which is the estimate for $\mathbb{P}\{T > 500\}$. Then we use the above formula to obtain $\mathbb{E}^\infty\{T\}$.

TABLE 1 shows the value of b suggested by theory for different ARLs and the value of b computed using the MC procedure described above. For comparison, we also obtain the ARL by treating this as a single subspace tracking problem for which PETRELS-FO is employed. These values of b are in parentheses.

To estimate the expected detection delay, we generate instances where the γ_t in the model has an abrupt jump Δ_γ at time $t = 200$.

$$\gamma_t = \begin{cases} 0.6 - \gamma_0 t & t = 1, 2, \dots, 199 \\ 0.6 - \Delta_\gamma - \gamma_0 t & t = 200, 201, \dots, 400 \end{cases} \quad (43)$$

Table 1: Average run length (ARL) $\mathbb{E}^\infty\{T\}$. In all cases the theoretical value is close to the MC estimate.

ARL	b	b MC with 0% data missing		b MC with 20% data missing		b MC with 40% data missing	
		MOUSSE	Single Subspace	MOUSSE	Single Subspace	MOUSSE	Single Subspace
1000	3.94	4.53	3.94	4.38	3.99	4.17	4.03
5000	4.35	5.87	4.64	5.38	4.73	5.02	4.75
10000	4.52	6.78	4.94	5.87	5.02	5.40	5.03

Then we apply the CUSUM statistics and find the expected detection delay $\mathbb{E}^0\{T\}$ with respect to $t = 200$ using 10000 Monte Carlo trials. We compare the expected detection delay of MOUSSE and the single subspace tracking method. Results corresponding to big ($\Delta_\gamma = 0.05$) and small ($\Delta_\gamma = 0.03$) magnitude of jump of γ_t are given in TABLE 2, 3 respectively. Again, values in parenthesis are obtained by using single subspace tracking. The threshold b 's are chosen according to the Monte Carlo results given in TABLE 1; *e.g.*, for the cell corresponding to ARL = 1000 and 0% missing data in TABLE 2 or 3, b should be set as 4.55 for MOUSSE and 4.28 for the single subspace method. TABLE 2, 3 demonstrate that MOUSSE has much smaller expected detection delay than a single subspace method.

Table 2: Detection delay when jump of γ_t is $\Delta_\gamma = 0.05$. Delays are significantly shorter with MOUSSE than with single subspace tracking.

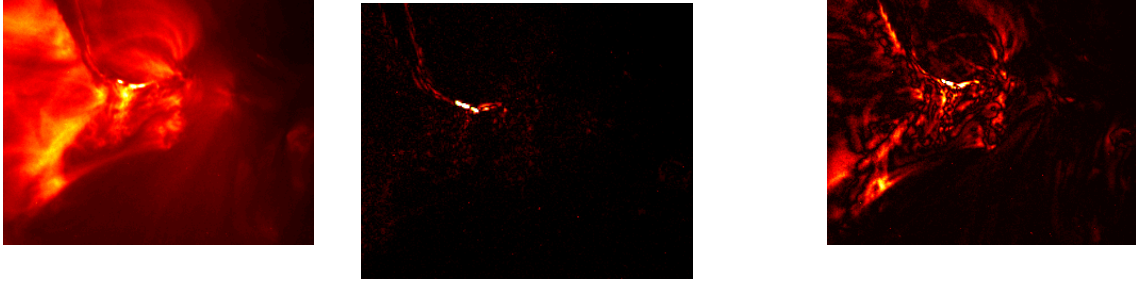
ARL	delay with 0% data missing		delay with 20% data missing		delay with 40% data missing	
	MOUSSE	Single Subspace	MOUSSE	Single Subspace	MOUSSE	Single Subspace
1000	1.92	60.26	1.85	56.99	1.88	53.71
5000	2.32	91.66	2.13	86.82	2.16	80.48
10000	2.68	104.91	2.31	98.57	3.31	91.21

Table 3: Detection delay when jump of γ_t is $\Delta_\gamma = 0.03$. Delays are significantly shorter with MOUSSE than with single subspace tracking.

ARL	delay with 0% data missing		delay with 20% data missing		delay with 40% data missing	
	MOUSSE	Single Subspace	MOUSSE	Single Subspace	MOUSSE	Single Subspace
1000	5.88	110.40	3.80	105.12	3.64	99.16
5000	6.68	163.80	5.46	157.52	5.04	147.66
10000	9.10	183.94	6.53	176.05	6.82	165.12

6.5 Real data

A video from the Solar Data Observatory, which demonstrates an abrupt emergence of a solar flare, can be found on <http://nislabs.ee.duke.edu/MOUSSE/index.html>. The Solar Object Locator for the original data is SOL2011-04-30T21-45-49L061C108. Also displayed is the residual \hat{e}_t of (26) obtained using MOUSSE, which clearly shows peaks in the vicinity of the solar flare. A frame from this dataset during a solar flare is shown in Figure 7a. The frame is of size 232×292 resulting in 67744 dimensional online data. To reduce difficulty of parameter tuning, we scale the pixel intensities in the dataset by multiplying the data by a factor of 10^{-4} to be consistent with the scale of our simulated data experiments. The parameters we use are $\epsilon = 0.3$, $\mu = 0.03$, and $\alpha = 0.85$. The video and the snapshots in Figure 7 demonstrate that MOUSSE can not only detect the emergence of a solar flares but also localize the flare by presenting \hat{e}_t , and these tasks are accomplished far more effectively with MOUSSE (even with $d = 1$) than with a single subspace. Note that with the single subspace tracking, the residual norm $e(t)$ is not a stationary time series prior to the flare and thus poorly suited for changepoint detection. In the original images, the background solar images has bright spots that are slowly and changing shape, which makes detection based on simple background subtraction incapable of detecting small transient flares. In contrast, with our approach, with K_t around 10, the underlying manifold structure is better tracked and thus yields more obvious error $e(t)$ when anomaly occurs.



(a) Snapshot of original SDO data at $t = 227$ (b) MOUSSE residual at $t = 227$ (c) Single subspace tracking residual at $t = 227$

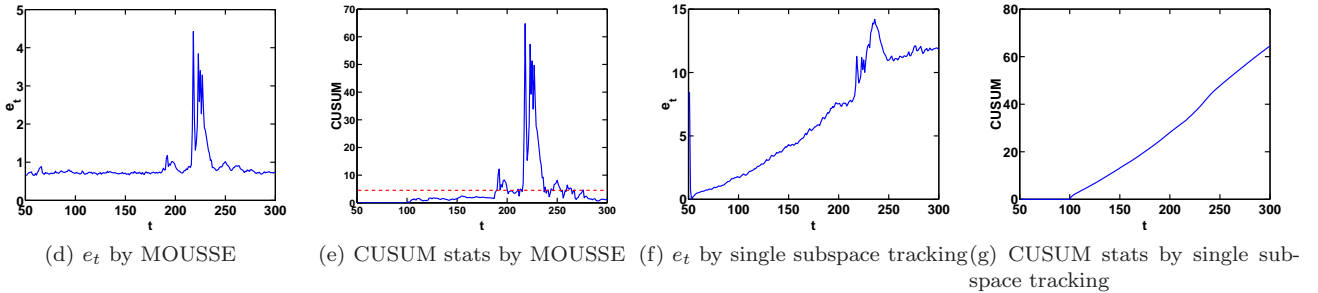


Figure 7: Detection of solar flare at $t = 227$: (a) snapshot of original SDO data at $t = 227$; (b) MOUSSE residual \hat{e}_t , which clearly identifies an outburst of solar flare; (c) single subspace tracking residual \hat{e}_t , which gives a poor indication of the flare; (d) $e(t)$ for MOUSSE which peaks near the flare around $t = 227$; (e) the CUSUM statistic (solid blue line) for MOUSSE with dashed red line indicating threshold b for $ARL=10000$; (f) $e(t)$ for single subspace tracking; (g) the CUSUM statistic for single subspace tracking. Using a single subspace gives much less reliable estimates of significant changes in the statistics of the frames.

Our second real data example is related to automatic identity theft detection. The basic idea is that consumers have typical spending patterns which change abruptly after identity theft. Banks would like to identify these changes as quickly as possible without triggering numerous false alarms. To test MOUSSE on this high-dimensional changepoint detection problem, we examined the E-commerce transaction history of people in a dataset used for a 2008 UCSD data mining competition at http://www.cs.purdue.edu/commugrate/data_access/all_data_sets_more.php?search. For each person in this dataset, there is a time series of transactions. For each transaction we have a 31-dimensional real-valued feature vector and a label of whether the transaction is “good” (0) or “bad” (1). The full dataset was generated for a generic anomaly detection problem, so it generally is not appropriate for our setting. However, some of these transaction timeseries show a clear changepoint in the labels, and we applied MOUSSE to these timeseries. In particular, we use MOUSSE to track the 31-dimensional feature vector and detect a changepoint, and compare this with the “ground truth” changepoint in the label timeseries. In calculating the CUSUM statistic, we estimate the μ_0 and σ_0 of equation 25 from e_1, \dots, e_{20} . After $t = 20$, every time the CUSUM statistic exceeds the threshold b and an changepoint is detected, we “reset” the CUSUM to only consider e_t after the most recently detected changepoint. This allows us to detect multiple changepoints in a timeseries.

The effect of our procedure for one person’s transaction history is displayed in Figure 8. We first see that MOUSSE accurately detects a temporally isolated outlier transaction at $t = 38$, after which the CUSUM is reset. After this, while MOUSSE does not generate particularly large spikes in e_t , the associated CUSUM statistic shows a marked increase near $t = 70$ and hits the threshold at $t = 72$, when the labels (not used by MOUSSE) change from 0 (good) to 1 (bad). After this the CUSUM is repeatedly reset and repeatedly detects the change in the statistics of e_t from the initial stationary process.

7 Conclusions

This paper describes a novel multiscale method for online tracking of high-dimensional data on a low-dimensional submanifold, and using the tracking residuals to perform fast and robust changepoint detection. Changepoint

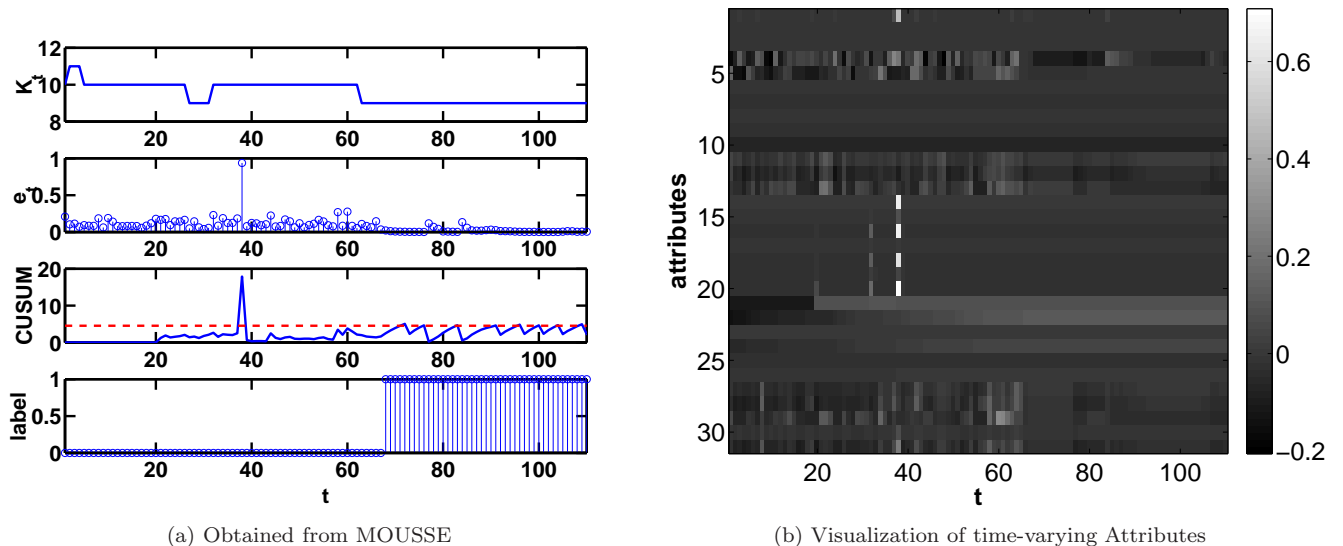


Figure 8: Credit card user data experiments. (a) From top to bottom: number of leaf nodes used by MOUSSE; e_t ; CUSUM statistic (solid blue line) and theoretical threshold b corresponding to $ARL = 10000$ (dashed red line); ground truth label. Note that the CUSUM statistic has a false alarm due to an outlier at $t = 38$, and it starts increasing at $t = 70$ and frequently hits the threshold afterwards due to the changepoint at $t = 70$. In this case CUSUM catches both the outlier and the changepoint. (b) Demonstration of the time-varying x_t (user attributes): each column corresponds to the 31-dimensional attribute vector at a given time. The white spots correspond to the outlier at time $t = 38$.

detection is an important subset of anomaly detection problems due to the ever-increasing volume of streaming data which must be efficiently prioritized and analyzed. The multiscale structure at the heart of our method is based on a Geometric MultiResolution Analysis which facilitates low-complexity piecewise-linear approximations to a manifold. The multiscale structure allows for fast updates of the manifold estimate and flexible approximations which can adapt to the changing curvature of a dynamic submanifold. These ideas have the potential to play an important role in analyzing large volumes of streaming data, which arises in remote sensing, credit monitoring, and network traffic analysis.

While the algorithm proposed in this paper has been focused on unions of subspaces, an important open question is whether similar techniques could be efficiently adopted based on sparse covariance matrix selection [51, 52]. The resulting approximation space may no longer correspond to a low-dimensional submanifold, but such structures provide good representations of high-dimensional data in many settings, and our future work includes tracking the evolution of a mixture of such structures. Issues related to non-Gaussian observation models, inverse problem settings, dynamical models, and optimal selection of the statistic used for changepoint detection (*i.e.*, alternatives to e_t , as considered in [53]) all pose additional interesting open problems.

A Proof of Theorem 1

Proof. From (1) and (9) we have

$$\beta = U^\top(v - c) + U^\top w, \quad (44)$$

Note that $U^\top w$ is zero-mean Gaussian random vector with covariance matrix $\sigma^2 U^\top U = \sigma^2 I$.

Next we consider the missing data case. Recall $\mathcal{P}_\Omega \in \mathbb{R}^{|\Omega| \times D}$ is a projection matrix. Define $w_\Omega = \mathcal{P}_\Omega w$. From (9) we have

$$\beta_\Omega = U_\Omega^\#(v_\Omega - c_\Omega) + U_\Omega^\# w_\Omega \quad (45)$$

Suppose in (1) we write $v - c = p + q$, with $p \in \mathcal{S}$ and $q \in \mathcal{S}^\perp$, where \mathcal{S}^\perp denotes the orthogonal subspace of \mathcal{S} .

Hence, $p = UU^\top(v - c)$ and $q = (I - UU^\top)(v - c)$. Let $p_\Omega = \mathcal{P}_\Omega p$, $q_\Omega = \mathcal{P}_\Omega q$. Hence, $v_\Omega - c_\Omega = p_\Omega + q_\Omega$. Note that

$$U_\Omega^\# p_\Omega = (U_\Omega^\top U_\Omega)^{-1} U_\Omega^\top \mathcal{P}_\Omega U U^\top (v - c) \quad (46)$$

$$= (U_\Omega^\top U_\Omega)^{-1} U_\Omega^\top U_\Omega U^\top (v - c) \quad (47)$$

$$= U^\top (v - c). \quad (48)$$

So

$$\beta_\Omega = U^\top (v - c) + U_\Omega^\# q_\Omega + U_\Omega^\# w_\Omega.$$

Hence

$$\begin{aligned} & \|\beta_\Omega - \beta\|^2 \\ & \leq 2\|U_\Omega^\# q_\Omega\|^2 + 2\|U_\Omega^\# w_\Omega - U^\top w\|^2 \\ & = 2\|(U_\Omega^\top U_\Omega)^{-1} U_\Omega^\top q_\Omega\|^2 + 2\|[(U_\Omega^\top U_\Omega)^{-1} U_\Omega^\top \mathcal{P}_\Omega - U^\top]w\|^2 \end{aligned}$$

We will bound these two terms separately.

First, note that

$$\|(U_\Omega^\top U_\Omega)^{-1} U_\Omega^\top q_\Omega\|^2 \leq \|(U_\Omega^\top U_\Omega)^{-1}\|_2^2 \|U_\Omega^\top q_\Omega\|^2 \quad (49)$$

where $\|A\|_2$ denotes the spectral norm of matrix A . Using [Lemma 2] in [49], we have that with probability $1 - \varepsilon$, if $|\Omega| \geq \frac{8}{3} d \text{coth}(U) \log(2d/\varepsilon)$,

$$\|U_\Omega^\top q_\Omega\|^2 \leq (1 + \theta)^2 \frac{|\Omega|}{D} \frac{d}{D} \text{coth}(U) \|q\|^2,$$

where $\theta = \sqrt{2 \frac{\max_{n=1}^D \|q_n\|^2}{\|q\|^2} \log(1/\varepsilon)}$. Using [Lemma 3] in [49] we have that provided that $0 < \ell < 1$, with probability at least $1 - \varepsilon$,

$$\|(U_\Omega^\top U_\Omega)^{-1}\|_2 \leq \frac{D}{(1 - \ell)|\Omega|}. \quad (50)$$

Combine these with (49), we have that with probability $1 - 2\varepsilon$,

$$\|(U_\Omega^\top U_\Omega)^{-1} U_\Omega^\top q_\Omega\|^2 \leq \frac{(1 + \theta)^2}{(1 - \ell)^2} \cdot \frac{d}{|\Omega|} \cdot \text{coth}(U) \|q\|^2. \quad (51)$$

Next we examine the noise term. Define

$$\tilde{w} = [(U_\Omega^\top U_\Omega)^{-1} U_\Omega^\top \mathcal{P}_\Omega - U^\top]w,$$

which is a zero-mean Gaussian random vector with covariance matrix

$$\Gamma = \sigma^2 (U_\Omega^\top U_\Omega)^{-1} - \sigma^2 I,$$

where we have used the fact that $\mathcal{P}_\Omega \mathcal{P}_\Omega^\top = I$. Hence we bound the tail of the noise power using Markov inequality:

$$\mathbb{P}(\|\tilde{w}\|^2 > 2\tau^2 \sigma^2) \leq e^{-\tau} \mathbb{E}\{e^{\|\tilde{w}\|^2/(2\tau\sigma^2)}\} \leq 2De^{-\tau} \quad (52)$$

provided that τ is sufficiently large such that the maximum eigenvalue is smaller than τ : $\lambda_{\max}((U_\Omega^\top U_\Omega)^{-1}) < \tau$, *i.e.*, $\tau > D/[(1 - \ell)|\Omega|]$, by noting that $\lambda_{\max}((U_\Omega^\top U_\Omega)^{-1}) = \|(U_\Omega^\top U_\Omega)^{-1}\|_2$. The last equality in (52) is because, under such condition:

$$\begin{aligned} & \mathbb{E}\{e^{\|\tilde{w}\|^2/(2\tau\sigma^2)}\} \\ & = \int e^{\|x\|^2/(2\tau\sigma^2)} (2\pi)^{-D/2} |\Gamma|^{-1/2} e^{-\frac{1}{2}x^\top \Gamma^{-1}x} dx \\ & = (2\pi)^{-D/2} |\Gamma|^{-1/2} \int e^{-\frac{1}{2}x^\top (\Gamma^{-1} - \tau^{-1}\sigma^{-2}I)x} dx \\ & = |\Gamma|^{-1/2} |\Gamma^{-1} - \tau^{-1}\sigma^{-2}I|^{-1/2} = |I - \tau^{-1}\sigma^{-2}\Gamma|^{-1/2} \\ & = |(1 + 1/\tau)I - \tau^{-1}(U_\Omega^\top U_\Omega)^{-1}|^{-1/2} \\ & \leq D[(1 + 1/\tau) - \tau^{-1}\|(U_\Omega^\top U_\Omega)^{-1}\|_2]^{-1/2} \\ & \leq D \left[(1 + 1/\tau) - \frac{D}{\tau(1 - \ell)|\Omega|} \right]^{-1/2} \\ & = D \left[1 - \frac{1}{\tau} \left(\frac{D}{(1 - \ell)|\Omega|} - 1 \right) \right]^{-1/2} \end{aligned} \quad (53)$$

In the last inequality, we have used (50). Note that $D/[(1-\ell)|\Omega|] > 1$, and the upper bound in (53) is smaller than $2D$ if $\tau > \frac{4}{3}(\frac{D}{(1-\ell)|\Omega|} - 1)$ or $\tau > \frac{4}{3}\frac{D}{(1-\ell)|\Omega|}$. Now we set $2De^{-\tau} = \varepsilon$, if ε is sufficiently small such that $\log(2D/\varepsilon) > \frac{4}{3}\frac{D}{(1-\ell)|\Omega|}$. Hence we have when $|\Omega| > \frac{4}{3}\frac{D}{(1-\ell)\log(2D/\varepsilon)}$, $\|\tilde{w}\|^2 < \frac{32}{9}\frac{D^2\sigma^2}{(1-\ell)^2|\Omega|^2}$ with probability $1 - \varepsilon$. Finally, combining (51) and (53), we obtain the statement in Theorem 1. \square

References

- [1] M. Basseville and I. V. Nikiforov, *Detection of abrupt changes: Theory and applications*, Prentice Hall, April 1993.
- [2] H. V. Poor and O. Hadjiladis, *Quickest detection*, Cambridge University Press, Dec. 2008.
- [3] A. Lakhina, M. Crovella, and C. Diot, “Diagnosing network-wide traffic anomalies,” in *Proc. of SIGCOMM*, 2004.
- [4] M. Raginsky, R. Willett, C. Horn, J. Silva, and R. Marcia, “Sequential anomaly detection in the presence of noise and limited feedback,” *IEEE Trans. Info. Theory*, vol. 58, no. 8, pp. 5544 – 5562, Aug. 2012.
- [5] K.-C. Lee and D. Kriegman, “Online learning of probabilistic appearance manifolds for video-based recognition and tracking,” in *Proc. of CVPR*, 2005, pp. 852 – 859.
- [6] M. Qu, F. Y. Shih, J. Jing, and H. Wang, “Automatic solar filament detection using image processing techniques,” *Solar Physics*, , no. 1-2, pp. 119–135, 2005, DOI: 10.1007/s11207-005-5780-1.
- [7] J. Kappenman, “A perfect storm of planetary proportions,” *IEEE Spectrum*, vol. 49, no. 2, pp. 26 – 31, Feb. 2012.
- [8] J. B. Tenenbaum, V. de Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.
- [9] S. Roweis and L. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 22, no. 5500, pp. 2323–2326, 2000.
- [10] M. Belkin, *Problems of Learning on Manifolds*, Ph.D. thesis, University of Chicago, 2003.
- [11] J. A. Costa and A. O. Hero, “Geodesic entropic graphs for dimension and entropy estimation in manifold learning,” *IEEE Trans. Sig. Proc.*, vol. 25, no. 8, pp. 2210–2221, 2004.
- [12] M. Belkin, P. Niyogi, and V. Singhwani, “Manifold regularization: A geometric framework for learning from labeled and unlabeled examples,” *J. of Machine Learning Res.*, vol. 7, pp. 2399 – 2434, 2006.
- [13] M. B. Wakin, “Manifold-based signal recovery and parameter estimation from compressive measurements,” submitted, 2009.
- [14] W. Allard, G. Chen, and M. Maggioni, “Multi-scale geometric methods for data sets II: Geometric multi-resolution analysis,” *App. and Comp. Harmonic Ana.*, vol. 32, no. 3, pp. 435 – 462, May 2011.
- [15] L. Balzano, R. Nowak, and B. Recht, “Online identification and tracking of subspaces from highly incomplete information,” in *Proc. Allerton Conf. on Comm., Control and Comp.*, Sept. 2010, pp. 704 – 711.
- [16] Y. Chi, Y. C. Eldar, and R. Calderbank, “PETRELS: Subspace estimation and tracking from partial observations,” in *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2012.
- [17] Y. Chi, Y. C. Eldar, and R. Calderbank, “Petrels: Parallel estimation and tracking of subspace by recursive least squares from partial observations,” *submitted to IEEE Trans. Sig. Proc., arXived.*, July 2012.
- [18] M. Kristan, D. Skočaj, and A. Leonardis, “Online kernel density estimation for interactive learning,” *Image Vision Comput.*, vol. 28, no. 7, pp. 1106–1116, 2010.
- [19] Ph. Rigollet and A. B. Tsybakov, “Linear and convex aggregation of density estimators,” *Math. Methods Statist.*, vol. 16, no. 3, pp. 260–280, 2007.

- [20] R. S. G. Mahapatruni and A. G. Gray, “Cake: Convex adaptive kernel density estimation,” *J. of Machine Learning Res. - Proceedings Track*, vol. 15, pp. 498–506, 2011.
- [21] John Lafferty and Larry Wasserman, “Rodeo: Sparse, greedy nonparametric regression,” *Ann. Stat.*, vol. 36, no. arXiv:0803.1709. IMS-AOS-AOS0318. 1, pp. 28–63, Mar 2008.
- [22] G. McLachlan and D. Peel, *Finite Mixture Models*, Wiley, 2000.
- [23] Q. Li, *Estimation of Mixture Models*, Ph.D. thesis, Yale University, 1999.
- [24] Q. Li and A. Barron, *Advances in Neural Information Processing Systems 12*, chapter Mixture Density Estimation, MIT Press, 2000.
- [25] G. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, Wiley, New York, 1997.
- [26] D. Bezier, “BAE to develop surveillance system,” *The Washington Post*, 2007, <http://www.washingtonpost.com/wp-dyn/content/article/2007/11/11/AR2007111101348.html>, retrieved 3-20-2012.
- [27] D. Hambling, “Special forces’ gigapixel flying spy sees all,” 2009, <http://www.wired.com/dangerroom/2009/02/gigapixel-flyin/>.
- [28] NASA, “SDO instruments,” Retrieved 7-30-2012, <http://sdo.gsfc.nasa.gov/mission/instruments.php>.
- [29] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, “Anomaly detection in crowded scenes,” in *IEEE Conf. on Comp. Vision and Pattern Rec., San Francisco*, 2010, pp. 1975 – 1981.
- [30] X. Hou and L. Zhang, “Saliency detection: A spectral residual approach,” in *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR07)*. *IEEE Computer Society*, 2007, pp. 1–8.
- [31] H. J. Seo and P. Milanfar, “Static and space-time visual saliency detection by self-resemblance,” *J. of Vision*, vol. 9, no. 12, pp. 1–27, Nov. 2009.
- [32] K. Kennedy, B. Mac Namee, and S. J. Delany, “Using semi-supervised classifiers for credit scoring,” 2012, <http://www.palgrave-journals.com/jors/journal/vaop/ncurrent/abs/jors201130a.html>.
- [33] T. Ahmed and M. Coates, “Multivariate online anomaly detection using kernel recursive least squares,” in *Proc. IEEE Infocom*, 2007.
- [34] A. S. Polunchenko, A. G. Tartakovsky, and N. Mukhopadhyay, “Nearly optimal change-point detection with an application to cybersecurity,” *submitted to Sequential Analysis*, Mar. 2012.
- [35] N. Patwari, A. O. Hero, III, and A. Pacholski, “Manifold learning visualization of network traffic data,” in *Proc. ACM SIGCOMM workshop on mining network data (MineNet)*, 2005, pp. 191 – 196.
- [36] D. Donoho, “Cart and best-ortho-basis selection: A connection,” *Annals of Stat.*, vol. 25, pp. 1870–1911, 1997.
- [37] Y. Chi, Y. C. Eldar, and R. Calderbank, “PETReLS: Subspace estimation and tracking from partial observations,” in *Int. Conf. on Acoustic, Speech, and Sig. Proc. (ICASSP)*, Mar. 2012.
- [38] K. Abed-Meraim, A. Chkeif, Y. Hua, and S. Attallah, “On a class of orthonormal algorithms for principal and minor subspace tracking,” *J. of VLSI Sig Proc.*, vol. 31, pp. 57–70, 2002.
- [39] R. Meir, “Nonparametric time series prediction through adaptive model selection,” *Machine Learning*, vol. 39, pp. 5–34, 2000.
- [40] L. Breiman, J. Friedman, R. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth, Belmont, CA, 1983.
- [41] R. Willett and R. Nowak, “Multiscale Poisson intensity and density estimation,” *IEEE Trans. Info. Theory*, vol. 53, no. 9, pp. 3171–3187, 2007.
- [42] A. Barron, J. Rissanen, and B. Yu, “Minimum description length principle in coding and modeling,” *IEEE Trans. Info. Theory*, vol. 44, no. 6, pp. 2743–2760, October 1998.

- [43] N. Merhav and M. Feder, “Universal prediction,” *IEEE Trans. Info. Theory*, vol. 44, no. 6, pp. 2124–2147, October 1998.
- [44] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, 4th edition, 2001.
- [45] E. S. Page, “Continuous inspection scheme,” *Biometrika*, vol. 41, no. 1/2, pp. 100 – 115, June 1954.
- [46] M. Pollak and D. Siegmund, “Sequential detection of a change in a normal mean when the initial value is unknown,” *Annals of Stats.*, vol. 19, no. 1, pp. 394 – 416, 1991.
- [47] D. Siegmund, *Sequential Analysis: Test and Confidence Intervals*, Springer, Aug. 1985.
- [48] Y. Xie and D. Siegmund, “Sequential multi-sensor change-point detection,” *submitted to Annals of Statis*, June 2012.
- [49] L. Balzano, B. Recht, and R. Nowak, “High-dimensional matched subspace detection when data are missing,” in *IEEE Int. Sym. Info. Theory*, June 2010, pp. 1638 – 1642.
- [50] D. Siegmund and E. S. Venkatraman, “Using the generalized likelihood ratio statistic for sequential detection of a change-point,” *Annals of Stat.*, vol. 23, no. 1, pp. 255 – 271, 1995.
- [51] A. d’Aspremont, O. Banerjee, and L. El Ghaoui, “First-order methods for sparse covariance selection,” *SIAM. J. Matrix Anal. & Appl.*, vol. 30, no. 56, 2008.
- [52] S. M. Kakade, O. Shamir, K. Sridharan, and A. Tewari, “Learning exponential families in high-dimensionals: Strong convexity and sparsity,” in *Proc. of Int. Conf. on Artificial Intel. and Stats*, 2010, pp. 381 – 388.
- [53] Y. Xie and D. Siegmund, “Parallel sequential multisensor changepoint detection,” in *Joint Stats. Meeting (JSM)*, San Diego, 2012.