Uniform Histograms for Change Detection in Multivariate Data

Giacomo Boracchi
Dipartimento di Elettronica,
Informazione e Bioingegeria
Politecnico di Milano, Italy
giacomo.boracchi@polimi.it

Cristiano Cervellera, Danilo Macciò
Institute of Intelligent Systems
for Automation
National Research Council, Italy
{cristiano.cervellera, danilo.maccio}@cnr.it

***Abstract*—We address change-detection problems in the challenging conditions where data are multivariate and no *a priori* information or experimental evidence suggests a specific family of distributions to match stationary data. In such nonparametric settings, one typically resorts to computing an empirical model for the distribution of stationary data in the form of histograms. We here analyze two ways for building histograms in the change-detection context. In particular, we consider histograms following a uniformity criterion: uniformity in the volume and uniformity in the density. In the former case the input domain is divided into a regular grid, while in the latter the input domain is adaptively partitioned to yield subsets having the same probability to contain stationary data. For both histograms we discuss nonparametric monitoring procedures which implement likelihood-based and distance-based approaches to detect changes in the distribution. In our experiments, performed both on synthetic and real-world datasets, we show that the combination of uniform density histograms and distance-based approaches achieves the best change-detection performance.**

***Index Terms*—Change detection, multivariate data, histograms, datastream, total variation distance.**

I. Introduction

Change detection is an important problem in many application domains, where changes might indicate an unforeseen evolution of the data-generating process or a fault in a sensor/machinery, to name a few examples. Change-detection problems are typically encountered in time-series analysis [1], quality inspection [2] and monitoring systems [3], [4]; in computational intelligence, change-detection tests are used to pilot the adaptation of learning systems in nonstationary environments (concept drift) [5], [6].

Unfortunately, in many practical situations, the forthcoming changes are *a priori* unknown. Change-detection problems are thus typically addressed by estimating the distribution of stationary data, namely ϕ_0, from a training set and then computing a suitable statistic to determine whether test data match ϕ_0 or not. There are two major approaches in the literature to take such a decision: *i*) determining how different the distributions of training and test data are (e.g., [7]) and *ii*) determining whether test data are likely to have been generated by ϕ_0 (e.g., [8], [9]). We can roughly distinguish these two approaches under the notions of *distance-based* and *likelihood-based*, respectively, even if there exist connections between the two (see, for instance, the discussion in [10]).

Implementing either a distance-based or a likelihood-based approach becomes particularly challenging when no *a priori* information or experimental evidence suggests that a specific family of distributions is able to properly approximate stationary data. In these unfortunate situations, which are rather common when input data are multivariate and possibly collect different sorts of measurements, it is not possible to learn ϕ_0 by simply fitting few parameters. Then, one has to resort to *nonparametric* methods, leading to an empirical characterization of ϕ_0.

One of the most popular and practical ways to obtain an empirical estimate of ϕ_0 consists in partitioning the input domain $\mathcal{X}$ into a union of disjoint subsets, and then estimating the probability for samples generated by ϕ_0 to fall inside each subset. In practice, this corresponds to building a *histogram* describing the distribution of stationary data over the given partition. In change-detection problems it is very important to choose the right partition, since this has relevant implications on the monitoring schemes that can be employed [11], [7]. However, to the best of the authors' knowledge, this aspect has never been extensively investigated in the related literature and, in particular, no comparison between the distance-based and the likelihood-based approaches in multivariate change detection has been performed.

We here investigate the use of histograms for change-detection purposes in multivariate data, and we provide a rigorous performance assessment of different algorithms implementing a distance-based or a likelihood-based approach. We consider histograms built upon partitions of $\mathcal{X}$ designed according to two different uniformity criteria: *i*) regular grids providing a *uniform volume* partition, and *ii*) adaptively defined subsets yielding a *uniform density* partition. We refer to both these as *uniform histograms*. We also describe efficient monitoring schemes to be adopted on uniform histograms, and discuss their main advantages and drawbacks. Our experiments, performed on both synthetically generated multivariate Gaussian data and on a 5-dimensional real-world dataset, indicate that distance-based algorithms are more effective at detecting changes than the likelihood-based ones, and that uniform density partitions are preferable.

The paper structure is as follows: Section II formulates the change-detection problem and defines the histograms.

Section III describes the likelihood-based and distance-based approaches to change detection, while Section IV describes uniform histograms, how these can be constructed and employed in change-detection algorithms. Our experiments are developed in Section V, while concluding remarks are in Section VI.

II. Problem Formulation

We consider a monitoring scenario where a d-dimensional stream is analyzed to detect changes in the distribution of the data-generating process. Namely, we assume that data in stationary conditions are i.i.d. realizations of a random variable that follows an unknown distribution ϕ_0. Our goal is to detect change points, namely those time instants τ when the distribution of the data-generating process changes:

$$\mathbf{x}_t \sim \begin{cases} \phi_0 & t < \tau \\ \phi_1 & t \geq \tau \end{cases}, \tag{1}$$

where $\mathbf{x}_t \in \mathcal{X} \subset \mathbb{R}^d$ denotes the data point observed at time t and $\phi_1 \neq \phi_0$ is the post-change data distribution. Model (1) corresponds to an abrupt and permanent change affecting ϕ_0. The input domain $\mathcal{X}$ is assumed to be a sufficiently large set to contain all the possible data points; if necessary, $\mathcal{X} = \mathbb{R}^d$.

We assume that both ϕ_0 and ϕ_1 are unknown, and only a training set $X = \{\mathbf{x}_t\}_{t=1,\dots,N}$ containing N stationary data is provided to estimate ϕ_0. In particular, we focus on situations where no analytic distribution seems to properly match the training data, so that ϕ_0 can not be fit on X e.g., through a maximum likelihood procedure. Thus, we resort to computing a histogram h^0 to estimate the distribution of stationary data, and use this for change-detection purposes. Our goal is to investigate monitoring schemes associated to uniform histograms (Section IV) and determine which is the most advantageous option in terms of change-detection performance.

A. Histograms

A histogram built from ϕ_0 consists in a partition of $\mathcal{X}$ into a collection of K subsets $\{S_k\}_{k=1,\dots,K}$, along with the associated probabilities $\{p_k^0\}_{k=1,\dots,K}$. More specifically, the histogram h^0 is defined as the collection of (subset, probability) pairs:

$$h^0(X) = \{(S_k, p_k^0)\}_{k=1,\dots,K}, \tag{2}$$

where p_k^0 corresponds to the probability for data generated by ϕ_0 to fall inside the corresponding subset S_k.

We require the subsets S_k to be a disjoint covering of $\mathcal{X}$, i.e., $\bigcup_{k=1}^K S_k = \mathcal{X}$ and $S_j \cap S_i = \emptyset$ for $j \neq i$. Each probability p_k^0 is computed by counting the number m_k of points of X belonging to S_k, i.e., $p_k^0 = m_k/N$.

B. Batch-wise Monitoring

We assume that data to be analyzed arrive in batches containing ν samples each, and we denote by W a batch of consecutive ν points. Changes are detected by operating batch-wise (as in [7]), assessing whether recent data in W are consistent with the reference histogram h^0, thus whether W contains a change point or data generated by ϕ_1. In a data-streaming scenario, batches could be conveniently selected as windows opened over the most recent data.

III. Monitoring Schemes

In what follows we survey the two major approaches behind monitoring schemes: the *likelihood-based* and *distance-based* one, and discuss how these are typically used in change-detection algorithms. Notice that, in the considered nonparametric settings, most of these monitoring schemes require, on top of the training set used to construct h^0, another set of stationary data for estimating the distribution of the adopted test statistics when the incoming batches W contain stationary data. In these cases, we must split X as $X = \{\mathbf{x}_t\}_{t=1,\dots,N^*} \cup \{\mathbf{x}_t\}_{t=N^*+1,\dots,N}$ and use only the first N^* data to build h^0.

From now on, we refer to "*computing the probabilities of W with respect to h^0*", to indicate computing the probabilities $\{p_k^W\}_{k=1,\dots,K}$ by counting how many points of W fall inside each subset S_k of h^0.

A. Likelihood-based Change Detection

Likelihood-based methods are widely adopted in the change-detection literature, see [8], [9] to name a few examples. These methods are very handy in those situations where the distribution of stationary data can be estimated by fitting a probability density function $\widehat{\phi}_0$ to X. According to the general monitoring scheme, the training phase consists in estimating $\widehat{\phi}_0$. Then, during operations, the likelihood of test data – or, to mitigate numerical problems, an approximated expression of the log-likelihood – is computed as:

$$\widehat{l}_t = \log\left(\widehat{\phi}_0(\mathbf{x}_t)\right) \quad \text{for all } t > N. \tag{3}$$

Since $\widehat{l}_t \in \mathbb{R}$, the change $\phi_0 \to \phi_1$ can be detected by tests designed for univariate streams, which are quite popular in the statistical process control literature [2]. Monitoring the log-likelihood is particularly advantageous since this propagates changes affecting the correlation among the components of ϕ_0, which are not perceivable when monitoring each component of $\mathbf{x}_t$ (separately) or its norm. Thus, the log-likelihood can be seen as an effective one-dimensional feature to be monitored for change-detection purposes. Moreover, it is reasonable to expect a change to shift (most probably decrease) the log-likelihood values, thus changes can be detected by simply monitoring the expectation of $\widehat{l}_t$ for $t > N$.

Likelihood-based monitoring schemes can be easily adopted in nonparametric settings, by straightforwardly replacing $\widehat{\phi}_0$ in (3) with h^0, namely

$$\widehat{l}_t(\mathbf{x}_t) = p_k^0 \quad \text{for all } t > N \tag{4}$$

being k the index of the subset S_k where $\mathbf{x}_t$ falls.

This yields a sequence $\{\widehat{l}_t\}_{t>N}$ of discrete and scalar data that can be in principle monitored in an element-wise and sequential manner. However, to fairly compare against

distance-based methods, we consider only *one-shot* monitoring techniques (Section II-B) that use an hypothesis test like the Lepage [12], Mann-Withney [13] or t-test [14] to determine whether the log-likelihoods computed from the batch W conform with training ones.

In particular, in our experiments we compute $\bar{l}_W$, the average log-likelihood over W, and compare it against $\bar{l}_0$, defined as the average log-likelihood over the training data $\{\mathbf{x}_t\}_{t=N^*+1,\dots,N}$ that were not used for estimating h^0. Such a comparison is performed using a t-test having as null hypothesis "the average log-likelihood over the test batch W equals $\bar{l}_0$", setting a desired confidence level α. We adopted a t-test since tests based on ranking statistics like the Mann-Withney [13] or Lepage [12] might fail on discrete data when these take few different values, a situation that may happen when monitoring discrete probabilities as in (4).

B. Distance-based Change Detection

As a general paradigm, distance-based methods detect changes by measuring a suitable distance between distributions. These methods typically compute, for each test batch W, the probabilities with respect to h^0 and compare these against $\{p_k^0\}_{k=1,\dots,K}$. Since probabilities in a histogram assume discrete values, such comparison has to be performed by means of distance measures between categorical distributions.

To this purpose we can employ the total variation [14], the Kolmogorov-Smirnov [14] or the Hellinger distance [7], to name a few examples. In particular, to determine whether the distribution of data in a test batch W matches h^0 it is possible to build empirical confidence regions for the chosen distance and estimate the p-values. The confidence region refers to the null hypothesis that no change has happened, thus it can be estimated computing the distribution of the considered distance when input batches are generated in stationary conditions.

In particular, we adopt the total variation distance, which in our context can be expressed as

$$d_{TV} = \frac{1}{2}\sum_{k=1}^{K} \left| p_k^0 - p_k^W \right| . \tag{5}$$

Thanks to its additive formulation, the total variation is less affected by errors in the estimated probabilities p_k^0 than other distances based on the maximum operator (such as the Kolmogorov-Smirnov distance), which can be instead seriously affected by large errors in few subsets of h^0. The distribution of d_{TV} in stationary conditions is estimated taking B non-overlapping batches of ν points from $\{\mathbf{x}_t\}_{t=N^*+1,\dots,N}$ (which were not used for estimating h^0, thus $B \leq \lfloor (N - N^*)/\nu \rfloor$), and computing for each batch W_b the probabilities $\{p_k^b\}_{k=1,\dots,K}$ with respect to h^0, for $b = 1,\dots,B$. These probabilities provide B different values for d_{TV} when no change has happened, which allows us to compute empirical confidence intervals and p-values for the test statistic in stationary conditions.

During operations, we compute the probabilities of each incoming batch W with respect to the reference h^0, to obtain the value of d_{TV} (5). The corresponding p-value determines whether to reject or not the null hypothesis, at a chosen confidence level α. When X does not contain enough points for building B independent batches having length ν, we can resort to a bootstrap procedure [15] from a single batch W.

Another viable option when comparing distributions is to employ a distance-based parametric test such as the Pearson chi-square test [14]. In this case, the test statistic for a batch W containing ν points has the form

$$d_P = \nu \sum_{k=1}^{K} \frac{(p_k^0 - p_k^W)^2}{p_k^0} \tag{6}$$

where $\{p_k^W\}_{k=1,\dots,K}$ is the set of probabilities computed from W with respect to h^0. Under the assumption that samples in W are generated from ϕ_0 and that p_k^0 corresponds to the true probability of ϕ_0 over S_k, d_P follows a χ^2 distribution with $K-1$ degrees of freedom. Thus, there is no need to estimate empirical p-values, and changes can be detected by a standard χ^2 test for goodness of fit [14] using (6) as the test statistic. In this case, the whole stationary training set X can be used to build h^0 (i.e., $N^* = N$).

In practice, Pearson chi-square test must be used with special care, since it is known that the distribution of d_P can be quite different from the theoretical χ^2 when the reference probabilities p_k^0 are estimated from data. This means that the results of the test might be unreliable when there are not enough training data, i.e., when we are not confident that all the subsets of h^0 contain a sufficient number of samples to allow an accurate estimation of the reference probabilities.

In contrast with likelihood-based monitoring schemes, which enable an element-wise analysis of the stream, distance-based ones can only operate batch-wise, since the whole batch is needed to estimate the distribution of test data. However, it is possible to use a sequential monitoring scheme on the values of d_{TV} (respectively d_P) computed over time.

IV. Uniform Histograms

We here describe the two schemes for partitioning the input domain following a uniformity criterion, over which the reference histogram h^0 is defined. We consider *i*) regular grids, which are quite traditional in the change-detection literature, and *ii*) uniform density partitions, for which we employ a recursive construction that we introduce next.

In both cases, the resolution of the partition $\{S_k\}_{k=1,\dots,K}$ is ruled by an integer parameter q that sets the number of intervals in which each dimension is divided, determining as a consequence the overall number of subsets K. The most apparent difference is that subsets S_k in regular grids are fixed, while in the uniform density case these are adaptively defined from the training set, each to include (approximatively) the same number of training samples (see Figures 1 and 2). The two partition schemes differ also in their actual construction and, most importantly, in their use within change-detection algorithms.

A. Regular grids

Regular grids enclose the training set X using q^d non-overlapping hyperectangles having the same volume. The partition includes also the complement of X to cover the whole domain $\mathcal{X}$, thus the overall number of subsets is $K = q^d + 1$. In particular, the grid is defined as follows.

- *Defining the grid boundaries*
  From the training set X we define a hyperrectangle Z that encloses all the training data. Thus,
  $$Z = \prod_{j=1}^{d} [(x)_j^{min}, (x)_j^{max}]$$
  where $(x)_j^{min}$ and $(x)_j^{max}$ denote the minimum and maximum value of the component j over the training set, respectively. Define $\bar{Z} = \mathcal{X} \setminus Z$ the complement of Z w.r.t. $\mathcal{X}$.
- *Defining rectangular subsets*
  For $j = 1, \ldots, d$ we divide the range of each component, namely, $\xi_j = (x)_j^{max} - (x)_j^{min}$ into q segments having the same length. This operation defines the regular grid dividing Z into q^d subsets of the form
  $$\prod_{j=1}^{d} [(x)_j^{min} + r_j \xi_j / q, (x)_j^{min} + (r_j + 1)\xi_j / q],$$
  with $r_j \in \{0, \ldots, q-1\}$.
- *Indexing subsets and computing probabilities*
  Associate an index to each of these subsets i.e., S_k from 1 to $K-1$. Compute the number m_k^0 of points of X falling into each subset S_k, thus set $p_k^0 = m_k^0 / N$. Define $S_K = \bar{Z}$ and set the probability $p_K^0 = 0$.

Notice that data in a test batch W might, in principle, fall outside Z, which is defined by the training set X. The subset S_K, having reference probability equal to 0, gathers all these points. For a batch W containing ν points, the corresponding probability is computed as $p_K^W = m_K^W / \nu$, where m_K^W is the number of points of W falling outside Z.

From the operational point of view, we can determine the subset containing any point $\mathbf{x} \in Z$ by iteratively searching through all the d components of $\mathbf{x}$. However, in regular grids this search can be solved analytically in a closed form expression. Let us assume, for the sake of simplicity and without loss of generality, that $Z = [0,1]^d$. Then, the center $\mathbf{c}_k$ of each subset S_k, $k \in \{1, \ldots, q^d\}$ in the grid can be univocally indexed as follows:

$$(c_k)_1 = \left\lfloor \frac{k-1}{q^{d-1}} \right\rfloor \frac{1}{q} + \frac{1}{2q} \tag{7}$$

and

$$(c_k)_i = \left\lfloor \frac{k^*_{i,d}}{q^{d-i}} \right\rfloor \frac{1}{q} + \frac{1}{2q} \tag{8}$$

where $(c_k)_i$ is the i-th component of $\mathbf{c}_k$, for $i = 1, \ldots, d$ and $k^*_{i,d} = \mod(k-1, q^{d-i+1})$. In practice, this indexing procedure corresponds to progressively scanning the set Z, sweeping all the dimensions in a raster style.

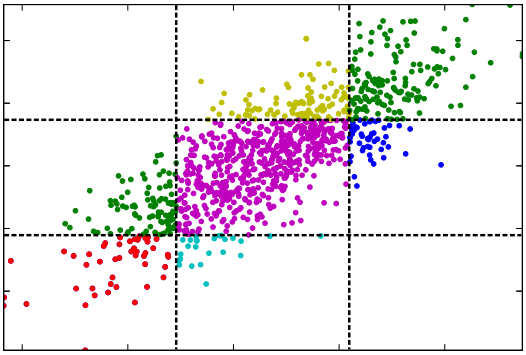

Fig. 1: Regular grid model for $d = 2$ and $q = 3$. All these subsets cover the same volume and have the same shape.

Thanks to this indexing, any point $\mathbf{x} \in [0,1)^d$ can immediately be mapped to the subset $\widehat{k}$ it belongs to by the following expression:

$$\widehat{k} = \lfloor (x)_1 q \rfloor \, q^{d-1} + \lfloor (x)_2 q \rfloor q^{d-2} + \cdots = \sum_{i=1}^{d} \lfloor (x)_i q \rfloor q^{d-i} \tag{9}$$

where $(x)_i$ denotes the i-th component of $\mathbf{x}$, while for $\mathbf{x} = (1, \ldots, 1)$ we set $\widehat{k} = q^d$. Figure 1 illustrates a grid partition obtained for a bivariate Gaussian with $q = 3$.

B. Uniform Density Partition

Regular grids partition the training set in subsets having equal volume. An alternative approach is to define the partition in such a way that all the subsets have the same probabilities. In other words, we look for a partition that yields a histogram corresponding to a set of categorical uniform probabilities, i.e., we want $p_k^0 \simeq 1/K$ for each $k = 1, \ldots, K$[1]. We propose a recursive scheme that splits all the subsets in the current partition along the same component. This procedure starts from the first component and stops once all the d components have been considered. Most importantly, such a recursive binary partition algorithm leads to a binary tree structure. The construction of such a tree can be summarized as follows.

- *Initialization*
  Given the training set X, we initially set $i = 1$ and define $v_1 = \mathcal{X}$ as the only subset of the partition. Then, we proceed as follows:
- While $i \leq d$ do:
  1) *Labeling the current tree subsets*
     Denote by K_i the current number of subsets in the partition and define the indexes of the subsets as $\{S_1, \ldots, S_{K_i}\}$.
  2) *Each subset is split along the i-th component in q subsets each containing the same number of points*

[1]The ' $\simeq$' symbol is due to the fact that it is not always possible to divide N points in K subsets, each containing exactly the same number of points.

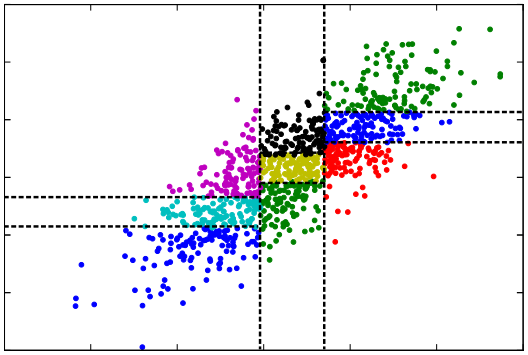

Fig. 2: Uniform density model for $d = 2$ and $q = 3$. All these subsets have the same probability to contain stationary data. The smaller subsets are placed over high-density regions.

For $k \in \{1, \dots, K_i\}$, consider the set $X_{i,k}$ containing all the values assumed by the i-th component of the points in S_k. Split S_k in q parts by cutting the i-th component along hyper-planes parallel to the other axes, and crossing the i-th axis at values corresponding to the $(r/q)\cdot 100$ percentiles of $X_{i,k}$, with $r = 1, \dots, q-1$.

3) *Cycle over all the components*
Set $i = i + 1$.

- *Labeling the final tree.*
The final partition $\{S_1, \dots, S_K\}$ is given by the subsets obtained after the last round of splits.
- *Compute the probabilities.*
For each subset S_k, compute m_k^0, the number of points of X falling into S_k, and set $p_k^0 = m_k^0/N$.

This procedure yields a total number of $K = q^d$ partition subsets, where each subset contains approximately N/K points. Figure 2 illustrates a uniform density partition for bivariate Gaussian data.

Thanks to the tree structure of such a histogram, locating any point $x \in X$ over the subsets $\{S_k\}_{k=1,\dots,K}$ is straightforward and very efficient. Thus, $\widehat{k}$ can be estimated by running a finite sequence of IF-THEN-ELSE statements.

C. Comments

As mentioned before, the main difference between the two partitions consists in how the concept of "uniformity" is implemented: in regular grids each subset has the same volume (Section IV-A), while in the uniform density case each subset approximatively contains the same number of training points (Section IV-B). There are however other important differences. While the number of subsets K in the considered histograms is certainly bound to an exponential growth with d, subsets in regular grids are mainly sparse, since the number of elements where p_k^0 is different from 0 is at most N. This is a key aspect from a computational point of view. In contrast, in uniform density histograms, all the $K = q^d$ subsets have non-zero probabilities, which however suggests that this partition can not be adopted when d is very large or q^d is larger than N. An advantage of uniform-density histograms is that they do not need to define a specific subset containing points falling outside Z, since here the subsets cover the whole $\mathcal{X}$.

Second, these two histograms can implement different types of change-detection tests. In fact, histograms built upon regular grids can be used both in likelihood-based tests (Section III-A) and in distance-based test (Section III-B). However, when using likelihood-based tests, remember that ranking statistics like the Mann-Withney [13] or the Lepage [12] might fail when probabilities p_k^0 might assume a limited number of values (e.g. when the histogram does not contain many samples and/or the probabilities are constant in some of cases). Distance-based monitoring schemes, which compute the empirical distribution of the statistics, like d_{TV} in (5) are also viable. In these cases it is not advisable to employ the Pearson chi-square test (6), since the reference probabilities might not be accurately estimated in subsets containing a very small number of points. Histograms built on uniform-density partition do not allow to pursue likelihood-based approaches since likelihood values in (4) would be always constant, disregarding the distribution of the data-generating process. Thus, the distance-based approach is the only viable option when using a uniform-density partition. In particular, since each subset contains approximately the same number of training points, it is possible to apply the Pearson chi-square test, making sure that there are enough training points falling in each subset S_k.

V. Experiments

In this section we present our experiments to assess the change-detection performance when using a histogram built upon a regular grid or a uniform density partition. In particular, we showcase the behavior of the change-detection monitoring schemes outlined in Section III using both synthetic and real-world datasets having dimensions ranging from $d = 2, \dots, 5$.

A. Datasets

1) Gaussian Dataset: We generate data from a known reference distribution ϕ_0 and introduce changes having a controlled magnitude. In particular, for $d = 2, \dots, 5$, we generate a multivariate Gaussian random variable having zero mean and covariance matrix Σ_0, i.e., $\phi_0 = \mathcal{N}(0, \Sigma_0)$. The post-change distribution is also a Gaussian $\phi_1 = \mathcal{N}(\mu_1, \Sigma_1)$, where the parameters μ_1 ad Σ_1 are defined by the algorithm in [16] to guarantee that the symmetric Kullback-Leibler divergence between ϕ_0 and ϕ_1 equals 1. As discussed in [16], controlling the change magnitude is very important when considering change-detection problems in different data dimensions.

In particular, we randomly define 1000 matrices Σ_0 and for each Σ_0 we define 100 different parameters (μ_1, Σ_1), thus each change $\phi_0 \to \phi_1$ corresponds to a rotation and translation of the stationary state. This setup eventually allows to test each change detection algorithm over a total of 10^5 different changes.

2) Real-Word Dataset: We consider the "Physicochemical Properties of Protein Tertiary Structure" dataset (*Protein* dataset in what follows) from the UCI repository [17], which contains 9 input features and more than 45.000 samples. These data appear to be far from being Gaussian, and it is difficult to fit any parametric distribution. Thus, this is the scenario where histograms are among the most attractive options for change-detection purposes. Unfortunately, we can not control the magnitude of the introduced change by [16] as this would require to fit a parametric model to the distribution of stationary data (e.g. a Gaussian mixture). Thus, we fix the dimension of the input to $d = 5$ selecting the first 5 components of the dataset.

Changes are introduced by translating the data, which corresponds to having a post-change distribution defined as $\phi_1(\cdot) = \phi_0(\cdot + \mathbf{v})$, where $\mathbf{v}$ is a randomly defined vector. In particular, we generate 1000 vectors $\mathbf{v}$ and use these to test all the considered change-detection algorithms.

B. Considered Change-Detection Algorithms

We consider change-detection algorithms built upon:

GRID-LB: histograms built upon regular grids using the likelihood-based test described in Section III-A.

GRID-TV: histograms built upon regular grids using a test on the p-values of d_{TV}.

TREE-TV: histograms yielding uniform density using a test on the p-values of d_{TV}.

TREE-PS: histograms yielding uniform density using the Person chi-square test, i.e. d_{PS}.

In addition to the aforementioned algorithms, only for the Gaussian dataset where ϕ_0 is known, we implement a likelihood-based parametric monitoring scheme that does not use any histogram and directly computes the likelihood in (3) with respect to the true distribution ϕ_0. Such ideal condition where ϕ_0 is known, which is seldom the case in practical applications, is considered as a reference in our experiments.

C. Experimental Setup

We repeat our experiments building histograms setting $q = 2$ and $q = 3$ to yield partitions having different resolutions both in regular grids and uniform density cases. Larger values of q would lead to histograms having too many subsets and, consequently, too few points inside each subset to guarantee accurate estimates when $d = 5$.

In the Gaussian dataset, for each change $\phi_0 \to \phi_1$, we generate a new training set X and batches W_b, while for the Protein dataset we randomly shuffle the whole dataset as in Section V-A to define every time a different training set. Batches are selected as non-overlaping windows containing ν points each form the shuffled dataset, and changes are introduced by summing a vector $\mathbf{v}$ to each component. All these change-detection algorithms have been configured to yield 5% type I errors (false positive rate).

The number N of training points in X is set to $1500 \cdot d$ in the Gaussian dataset, and equal to 15000 for the Protein dataset, which requires a larger training set to estimate a more complex distribution. In both cases, test batches W contain $\nu = 150$ points, and we use $B = 100$ stationary batches to compute the empirical distribution of d_{TV}. These settings yield reliable estimates of the empirical distributions such that the false positive rate is close to 5% with all the datasets and algorithms.

TABLE I: Median of the powers for Gaussian data ($q = 2$).

	$d = 2$	$d = 3$	$d = 4$	$d = 5$
GRID-TV	0.99	0.86	0.62	0.33
TREE-TV	1	0.97	0.81	0.46
TREE-PS	1	0.98	0.85	0.49
GRID-LB	0.3	0.14	0.09	0.08

TABLE II: Median of the powers for Gaussian data ($q = 3$).

	$d = 2$	$d = 3$	$d = 4$	$d = 5$
GRID-TV	1	0.89	0.56	0.26
TREE-TV	1	0.99	0.79	0.32
TREE-PS	1	1	0.86	0.44
GRID-LB	0.31	0.11	0.08	0.07

TABLE III: PAR-LB: median of the powers for Gaussian data.

	$d = 2$	$d = 3$	$d = 4$	$d = 5$
PAR-LB	0.86	0.73	0.61	0.51

TABLE IV: Median of the powers for the *Protein* dataset.

	GRID-TV	TREE-TV	TREE-PS	GRID-LB
q=2	0.11	0.87	0.91	0.08
q=3	0.16	0.96	0.98	0.07

Change-detection performance is assessed by the empirical power of each test, computed as the number of correct detections over 100 batches containing changed data. The boxplots in Figure 3 portray the distribution of test powers for the 10^5 considered changes $\phi_0 \to \phi_1$ in the Gaussian case, when increasing data dimension d. The medians of these boxplots are reported in Tables II. All these values refer only to the partitioning built setting $q = 3$, while boxplots corresponding $q = 2$ exhibit a similar behavior and have not been reported. The medians of the test powers when $q = 2$ are reported in Table I. For a reference comparison, Figure 3 and Table III report also the powers of the PAR-LB algorithm.

The boxplots of the test powers computed from the Protein dataset are depicted in Figure 4 for $q = 3$. Again, the boxplots for the case $q = 2$ show a similar behavior and the medians of the power are reported in Table IV for both $q = 2$ and $q = 3$.

D. Results And Discussion

All the results consistently indicate two important outcomes: first, distance-based algorithms outperform likelihood-based ones. This point obviously concerns the considered batch-wise monitoring scheme, where our experiments show that comparing the empirical distribution of the input and stationary

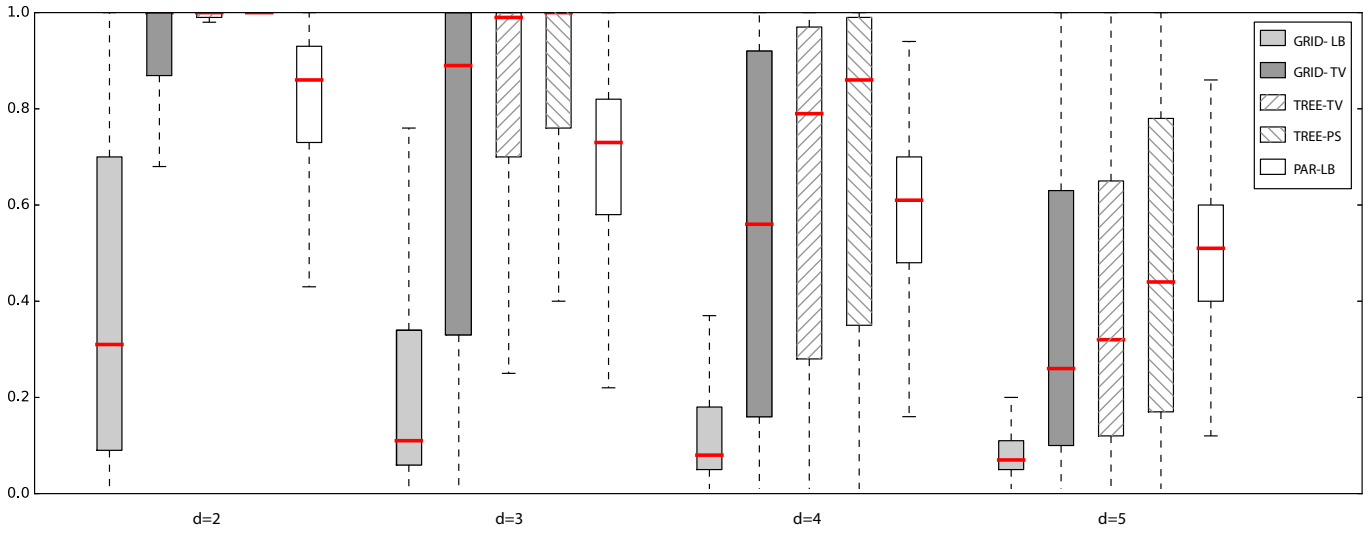

Fig. 3: Boxplots of all the considered methods in the Gaussian dataset. Change-detection algorithms implementing a minimum-distance monitoring scheme (GRID-TV, TREE-TV, TREE-PS) outperform those that monitor the likelihood (GRID-LB, PAR-LB). In particular, distance-based monitoring schemes implemented on the uniform-density partitions (TREE-TV, TREE-PS) achieve the best results.

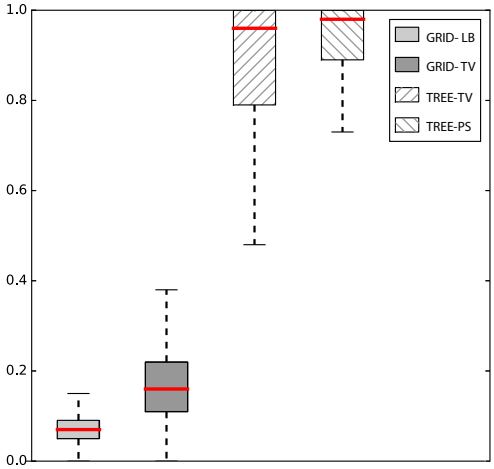

Fig. 4: Boxplots of the powers for the *Protein* dataset when $q = 3$. The change-detection performance is in line with the Gaussian case (Figure 3), and in particular distance-based methods implemented on the uniform-density partitions (TREE-TV, TREE-PS) achieve the best results.

data turns to be more effective than monitoring variations in the expectation of the log-likelihood. The PAR-LB test, which as expected outperforms its empirical counterpart (GRID-LB), achieves best power levels that are inferior to those of distance-based algorithms. This indicates that changes can be better detected by analyzing the data distribution on the whole input domain rather than simply testing whether the average log-likelihood has changed by a t-test. As expected, the boxplots of the PAR-ML algorithm exhibit also a smaller dispersion than all the other algorithms since this parametric method does not suffer from density-estimation errors.

The second outcome is that the best option among distance-based monitoring schemes consists in employing histograms yielding uniform density. Moreover, our experiments show that the TREE-TV and the TREE-PS algorithms are the only ones achieving good performance in the Protein dataset. In this real-world dataset, the grid provides a very unbalanced covering of the data, since most of the subsets turn out to contain too few points to allow a reliable estimate of the reference distribution. In contrast, the uniform density partition does not suffer of this problem since it adapts to the shape of the distribution.

The boxplots of the test power for the Gaussian case indicate there is quite a substantial variability in these results also due to the wide range of considered changes. However, paired t-tests performed on these powers confirm that their means are statistically different.

As a final remark, we comment that on Gaussian data there is a consistent decay of the test power as the dimension d increases. There are two reasons for such a performance decay. First, when d increases, estimating h^0 becomes more difficult and requires more data. Second, the *detectability loss* phenomenon, which was investigated in [18] for likelihood-based approaches and analytically proved for Gaussian data, indicates that change detection becomes intrinsically more challenging. Detectability loss explains the performance decay of PAR-ML, which does not have to estimate h^0. Figure 3 indicates that detectability loss occurs also when adopting histograms and distance-based approaches, and not only likelihood-based methods.

Concerning the resolution parameter q, in the Gaussian case $q = 2$ (i.e., lower resolution) appears to be better when $d = 4, 5$, while $q = 3$ works better when $d = 2, 3$. This is probably due to the fact that setting $q = 3$ leads to subsets containing not enough points from the training set when $d = 4, 5$. This problem affects also the TREE-PS algorithm since the Pearson test relies on the assumption that the reference uniform probability in the tree subsets is estimated accurately.

VI. Conclusions

In this work we have investigated the use of histograms built according to two uniformity criteria for modeling the distribution of stationary data in change-detection problems. We have considered both histograms constructed on a regular grid, yielding subsets having uniform volume, and histograms obtained through a recursive procedure which yields subsets having uniform density. We have also discussed how the choice of the histogram influences the design of change-detection algorithms implementing distance-based and likelihood-based approaches. Our experiments show that the best option in batch-wise monitoring consists in combining uniform density histograms with a distance-based method.

Future work aims at developing further uniform density histograms for high-dimensional change-detection problems [19], where it is not feasible to grow the partition size as q^d. Moreover, we will extend our experimental assessment to include other nonparametric monitoring schemes such as kernel-density estimation methods used for learning in nonstationary environments [20], [21].

References

[1] G. J. Ross, D. K. Tasoulis, and N. M. Adams, "Nonparametric monitoring of data streams for changes in location and scale," *Technometrics*, vol. 53, no. 4, pp. 379–389, 2011.

[2] M. Basseville and I. V. Nikiforov, *Detection of abrupt changes: theory and application*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993.

[3] M. Moshtaghi, C. Leckie, S. Karunasekera, and S. Rajasegarar, "An adaptive elliptical anomaly detection model for wireless sensor networks," *Computer Networks*, vol. 64, pp. 195 – 207, 2014.

[4] C. Alippi, G. Boracchi, and B. Wohlberg, "Change detection in streams of signals with sparse representations," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2014, pp. 5252 – 5256.

[5] C. Alippi, G. Boracchi, and M. Roveri, "Just-in-time classifiers for recurrent concepts," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 4, pp. 620–634, April.

[6] J. a. Gama, I. Zliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 44:1–44:37, Mar. 2014.

[7] G. Ditzler and R. Polikar, "Hellinger distance based drift detection for nonstationary environments," in *Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2011 IEEE Symposium on*, April 2011, pp. 41–48.

[8] L. I. Kuncheva, "Change detection in streaming multivariate data using likelihood detectors," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 5, 2013.

[9] X. Song, M. Wu, C. Jermaine, and S. Ranka, "Statistical change detection for multi-dimensional data," in *Proceedings of International Conference on Knowledge Discovery and Data Mining (KDD)*, 2007.

[10] C. R. Blyth, "On the inference and decision models of statistics," *Ann. Math. Statist.*, vol. 41, no. 3, pp. 1034–1058, 1970.

[11] R. Sebastião, J. Gama, P. P. Rodrigues, and J. Bernardes, "Monitoring incremental histogram distribution for change detection in data streams," in *Knowledge Discovery from Sensor Data. Lecture Notes in Computer Science vol. 5840*, M. M. Gaber, R. R. Vatsavai, O. A. Omitaomu, J. Gama, N. V. Chawla, and A. R. Ganguly, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 25–42.

[12] Y. Lepage, "A combination of Wilcoxon's and Ansari-Bradley's statistics," *Biometrika*, vol. Vol. 58, No. 1, pp. 213–217, April 1974.

[13] H. B. Mann and D. R. Whitney, "On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other," *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, 1947.

[14] E. L. Lehmann and J. P. Romano, *Testing Statistical Hypotheses*. New York, NY: Springer Science+Business Media, Inc. Springer e-books, 2005.

[15] T. Hastie, R. Tibshirani, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations*. Springer-Verlag, New York, February 2009.

[16] C. Alippi, G. Boracchi, and D. Carrera, *CCM: Controlling the Change Magnitude in High Dimensional Data*. Cham: Springer International Publishing, 2016, pp. 216–225.

[17] K. Bache and M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[18] C. Alippi, G. Boracchi, D. Carrera, and M. Roveri, "Change detection in multivariate datastreams: Likelihood and detectability loss." in *in Proc. 2016 International Joint Conference on Artificial Intelligence 2016*, July 2016, pp. 1368–1374.

[19] A. Zimek, E. Schubert, and H.-P. Kriegel, "A survey on unsupervised outlier detection in high-dimensional numerical data," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 5, no. 5, 2012.

[20] K. B. Dyer, R. Capo, and R. Polikar, "Compose: A semisupervised learning framework for initially labeled nonstationary streaming data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, 2014.

[21] G. Krempl, "The algorithm APT to classify in concurrence of latency and drift," in *Advances in Intelligent Data Analysis X (IDA)*, 2011, pp. 222–233.