A pdf-Free Change Detection Test Based on Density Difference Estimation

Li Bu, *Student Member, IEEE*, Cesare Alippi, *Fellow, IEEE*, and Dongbin Zhao, *Senior Member, IEEE*

***Abstract*— The ability to detect online changes in stationarity or time variance in a data stream is a hot research topic with striking implications. In this paper, we propose a novel probability density function-free change detection test, which is based on the least squares density-difference estimation method and operates online on multidimensional inputs. The test does not require any assumption about the underlying data distribution, and is able to operate immediately after having been configured by adopting a reservoir sampling mechanism. Thresholds requested to detect a change are automatically derived once a false positive rate is set by the application designer. Comprehensive experiments validate the effectiveness in detection of the proposed method both in terms of detection promptness and accuracy.**

***Index Terms*— Concept drift, least squares density-difference (LSDD)-based method, probability density function (pdf)-free, three-level threshold mechanism.**

I. Introduction

THE traditional learning framework assumes the stationary hypothesis for the process generating the data, implying that its statistical characterization does not change with time. However, such a hypothesis constitutes a first-order approximation of the reality and is hardly met in those applications where time variant phenomena affect the environment, the sensors acquiring the data streams or both.

The literature addressing learning in nonstationary or evolving environments classifies existing methods as passive or active depending on the learning mechanism adopted to deal with the process evolution [1]. We say that the approach is passive when the application undergoes a continuous learning without knowing whether changes in stationarity occurred or not [2], [3]. Differently, within an active approach, a triggering mechanism, e.g., a change detection test (CDT) [4], [5], is considered and the application evolves and adapts to track the evolution of the environment only after a change is detected. In this paper, we focus on changes in stationarity—or concept drift—and assume that data streams to be inspected are possibly infinite sequences of independent and identically distributed (i.i.d.) samples drawn from a random variable following an unknown continuous probability density function (pdf) $p(x)$. Suitable transformations should be applied to data streams when facing with signals and nonrandom variables in order to meet the i.i.d. hypothesis, e.g., as done in [6].

The rich literature presents many solutions for detecting concept drift within an unsupervised framework (i.e., only changes affecting the pdf of inputs are considered), mostly by observing statistical features extracted from the pdf, e.g., mean, variance, or the time evolution of other statistics. For instance, Hawkins and Zamba [7] introduced a change-point formulation by inspecting the mean and/or variance in normally distributed data streams with a single chart. A sequential estimation technique was proposed in [8], which simultaneously monitors data streams and updates the method parameters. Ross *et al.* [9], [10] extended those works to allow for fully nonparametric detections in non-Gaussian sequences. Nonparametric hypothesis tests, such as the Mann–Whitney test [11], the Mood test [12] and the Lepage one [13], have been advanced in [9] to create a streaming change-point model (CPM) able to deal with arbitrary continuously distributed univariate data streams; two control charts based on Kolmogorov–Simirnov [14] and Cramer–von Mises tests [15] have been proposed in [10] to detect arbitrary changes without assuming a known distribution. Therefore, authors adopt the average run length (ARL0) index defined as the average number of observations between consecutive false detections; the index provides the first statistical moment for the occurrence of false positives (FPs). Requested thresholds are obtained with a Monte Carlo analysis once a predefined ARL0 value has been set. Raza *et al.* [16], [17] proposed an exponentially weighted moving average (EWMA)-based method to monitor autocorrelated observations, which is suitable for real-time adaptive classification problems. Alippi *et al.* [18]–[20] proposed and advanced the just-in-time framework where applications are automatically adapted following a detected change. The CI-CUSUM test [18] and ICI-based CDT [21] can detect trends and drifts without any prior by relying on extracted features, with requested thresholds estimated directly from the training set. Kuncheva and Faithfull [22] proposed a feature extraction method by applying principal component analysis for change detection in multidimensional data, arguing that the least important components are more sensitive to changes. Most of these methods operate on scalar streams; extension to the multidimensional case is mostly implemented by inspecting each dimension independently followed by a final consensus method aggregating provided results.

Some authors follow a different philosophy for concept drift detection by comparing two pdfs; one assumed to represent

Manuscript received May 24, 2016; revised August 25, 2016; accepted October 18, 2016. This work was supported by the National Natural Science Foundation of China under Grant 61273136, Grant 61573353, Grant 61533017, and Grant 61603382.

L. Bu and D. Zhao are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, University of Chinese Academy of Sciences, Beijing 100190, China (e-mail: bulipolly@gmail.com; dongbin.zhao@ia.ac.cn).

C. Alippi is with the Politecnico di Milano, 20133 Milano, Italy and Università della Svizzera italiana, 6904 Lugano, Switzerland (e-mail: cesare.alippi@polimi.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2016.2619909

the stationary case, and the other associated with data possibly affected by changes. In this direction, Schilling [23] proposed a k-nearest neighbor (KNN)-based test measuring the proportion of observations and their KNNs belonging to the same sample. The statistic asymptotically satisfies a normal distribution under some mild assumptions, and is compared with a predefined threshold to detect a change.

Many authors have proposed solutions to detect changes in classification problems (supervised methods). Therefore, the distribution of outputs or the conditional probability is inspected for changes, sometimes in addition to inspection of changes in the distribution of inputs [24]. Since we assume the pdf to be continuous, we are not addressing changes here, affecting the output of a classification system; in such a case, suitable methods must be considered to complete the investigation for concept drift. For instance, methods proposed in [25]–[27] aim at detecting changes at the classification error level, which satisfies a Bernoulli distribution. The EWMA-based method proposed in [25] can detect changes under a controlled FP rate; the SeqDrift1 [26] applies the Bernstein inequality to measure the deviation degree of the two mean values and SeqDrift2 [27] extends such a work using reservoir sampling to achieve low FP rates and reduce detection delay.

Only few papers address the detection challenge directly at the pdf level, here supposed to be continuous. These methods do not specify the type of detected changes, since any change affecting the pdfs will be detected provided change magnitudes are large enough. However, the major difficulty here is associated with the generally limited data set, situation that prevents any effective estimate of the pdf. Dasu *et al.* [28] proposed an information-theoretic approach, which utilizes kdq-trees to form the empirical distributions starting from two independent data sets representative of the stationary and the—possibly changed—distributions. Comparison between the two estimates is then implemented with the Kullback-Leibler-divergence. Sugiyama *et al.* proposed instead to estimate density ratio [29], [30] or density difference [31], [32] of the two subsets directly with Gaussian kernel functions. These methods overcome the drawback of the traditional two-step procedures requiring at first to estimate the two densities, operation that amplifies errors. The density-ratio and density-difference approaches measure the dissimilarity of the two pdfs: the higher the obtained value, the larger the difference. Density difference approaches are desirable, since derived values are finite provided that each density is bounded, whereas density ratios might diverge to infinity even under mild conditions [31], e.g., when distributions are Gaussians. However, neither in above research a reasonable threshold is given to detect a change, nor the FP rate is controllable by the application designer. Another major aspect is that derived estimates are very sensitive to both subset size and distribution, which makes it hard to derive a suitable threshold.

The above-mentioned approaches show to be very attractive and worth further investigation. In this paper, we propose a pdf-free CDT based on the least squares density-difference (LSDD) estimation method. The test shows to be particularly effective in multidimensional streaming applications. Considering the high variance of LSDD values for a given subset (window) size, a bootstrapping procedure has been presented to characterize such a variability; the exercise becomes particularly relevant in those scenarios encompassing small data sets. We then derive thresholds for change detection from the bootstrapped-based distribution in stationary conditions, which satisfy user-defined tolerated FP rates. In order to be sensitive to small changes, i.e., reduce false negatives (FNs), a three-level threshold mechanism is proposed with three detection options: provide a warning, confirm a change, or clear the warning. To the best of our knowledge, what here proposed is the first density-difference-based CDT:

1) introducing a control of FPs;
2) presenting a reservoir sampling mechanism to update the reference window permitting the method to become immediately operational without requiring a huge training set;
3) proposing a three-level threshold mechanism to reduce FNs through an increased detection sensitivity with thresholds associated with designer-tunable FP rates.

The structure of this paper is as follows. Section II presents the LSDD approach. Section III introduces the threshold mechanism with thresholds chosen to satisfy a predefined FP rate. The proposed LSDD-based CDT is given in Section IV, as with the hierarchical threshold mechanism designed to improve detection sensitivity. In Section V, comprehensive experiments validate the effectiveness of our proposed method. Section VI concludes the paper.

II. LSDD Method

The density-difference estimation method [32] aims at measuring the LSDD

$$D^2(p,q) = \int (p(x) - q(x))^2 dx \tag{1}$$

where $x \in R^d$ is a real vector, and $p(x)$ and $q(x)$ are two continuous pdfs. Within a change detection framework pdfs, $p(x)$ and $q(x)$ refer to the prechange and a possible postchange condition, respectively. Since both $p(x)$ and $q(x)$ are unknown, we estimate their difference $p(x) - q(x)$ with a linear-in-parameters Gaussian kernel function

$$g(x,\Theta) = \Theta^T K = \sum_{i=1}^{2n} \theta_i \exp\left(-\frac{||x - c_i||_2^2}{2\sigma^2}\right) \tag{2}$$

where $\Theta = (\theta_1, \ldots, \theta_i, \ldots, \theta_{2n})$ is the parameters vector, K is the Gaussian kernel vector, and $2n$ is the number of considered kernel functions. The kernel centers are chosen as $(c_1, \ldots, c_{2n}) = (x_{p,1}, \ldots, x_{p,n}, x_{q,1}, \ldots, x_{q,n})$ and are representative of pdfs $p(x)$ and $q(x)$, in the sense that n samples are drawn from $p(x)$ and n from $q(x)$. Scaling parameter σ is chosen during the training phase as the median distance between points in the aggregate sample as recommended in [33] $\sigma = \text{median}(||x_i - x_j||_2, 0 < i < j \leq N_t)$, where N_t is the cardinality of the training set.

The optimal parameter Θ^* is obtained by minimizing the squared loss

$$J(\Theta) = \int (g(x,\Theta) - (p(x) - q(x)))^2 dx. \tag{3}$$

By adding an $L2$-regularizer to (3) to request a smooth solution, the optimization problem becomes

$$\begin{aligned}\hat{\Theta} &= \arg\min_{\Theta} (J(\Theta) + \lambda \Theta^T \Theta) \\ &= \arg\min_{\Theta} (\Theta^T H \Theta - 2h^T \Theta + \lambda \Theta^T \Theta)\end{aligned} \tag{4}$$

where $\lambda \geq 0$ is the regularization parameter, H is a $2n \cdot 2n$ matrix, and h is a $2n \cdot 1$ vector. Defined as $H_{i,j}$ the generic component of matrix H and h_i the ith component of the vector h, we have that

$$\begin{aligned}H_{i,j} &= \int \exp\left(-\frac{||x-c_i||_2^2}{2\sigma^2}\right)\exp\left(-\frac{||x-c_j||_2^2}{2\sigma^2}\right)dx \\ &= (\pi\sigma^2)^{d/2}\exp\left(-\frac{||c_i-c_j||_2^2}{4\sigma^2}\right)\end{aligned} \tag{5}$$

$$\begin{aligned}h_i &= \int \exp\left(-\frac{||x-c_i||_2^2}{2\sigma^2}\right)p(x)dx \\ &\quad - \int \exp\left(-\frac{||x-c_i||_2^2}{2\sigma^2}\right)q(x)dx\end{aligned} \tag{6}$$

$i, j = 1, \ldots, 2n$. Since $p(x)$ and $q(x)$ are unknown pdfs, $\hat{h}_i$ is estimated with Monte Carlo on two data subsets: those associated with $p(x)$ (the pdf of the reference window) and those associated with the testing window to be checked for the presence of a change and associated with $q(x)$

$$\begin{aligned}\hat{h}_i &= \frac{1}{n}\sum_{j=1}^{n}\exp\left(-\frac{||x_{p,j}-c_i||_2^2}{2\sigma^2}\right) \\ &\quad -\frac{1}{n}\sum_{j=1}^{n}\exp\left(-\frac{||x_{q,j}-c_i||_2^2}{2\sigma^2}\right).\end{aligned} \tag{7}$$

Finally, $\hat{\Theta}$ can be expressed as

$$\hat{\Theta} = (H + \lambda I)^{-1}\hat{h}. \tag{8}$$

By replacing $p(x) - q(x)$ with $g(x, \hat{\Theta})$ in (1), the D^2-distance can be estimated by two equivalent expressions

$$\begin{aligned}\hat{D}_1^2(p,q) &= \int g(x,\hat{\Theta})^2 dx = \int (\hat{\Theta}^T K)^2 dx \\ &= \hat{\Theta}^T \left(\int K K^T dx\right)\hat{\Theta} \\ &= \hat{\Theta}^T H \hat{\Theta}\end{aligned} \tag{9}$$

$$\begin{aligned}\hat{D}_2^2(p,q) &= \int g(x,\hat{\Theta})p(x)dx - \int g(x,\hat{\Theta})q(x)dx \\ &= \hat{\Theta}^T h \approx \hat{\Theta}^T \hat{h}.\end{aligned} \tag{10}$$

Sugiyama *et al.* [31] combined them together to reduce the bias contribution brought by λ as

$$\hat{D}^2(p,q) = 2\hat{\Theta}^T\hat{h} - \hat{\Theta}^T H \hat{\Theta} \tag{11}$$

and $2\hat{\Theta}^T\hat{h} - \hat{\Theta}^T H\hat{\Theta} \geq \hat{\Theta}^T\hat{h} \geq \hat{\Theta}^T H\hat{\Theta} \geq 0$.

When $\lambda = 0$, the optimization problem reduces to the original one and no overfitting control is introduced; when λ is large, the approximating model is smooth, and the $\hat{D}^2$ values are generally small. The optimal parameter λ can be estimated with cross validation to minimize the squared loss, as done in [31] and [32] where the pdfs $p(x)$ and $q(x)$ are different. Whenever we compare two subsets drawn from a stationary distribution, which is the case both in the training phase and when the testing window associated with $q(x)$ refers to stationary data [$q(x)$ is identical to $p(x)$], cross validation does not work well. In fact, the density difference is so small that a large λ associated with a smooth fitting is always chosen, and the estimated LSDD values in a nonstationary data set diverge a lot from their real values.

Here, we introduce the relative difference (RD)

$$\begin{aligned}RD &= \frac{\hat{\Theta}^T\hat{h} - \hat{\Theta}^T H\hat{\Theta}}{\hat{\Theta}^T\hat{h}} \\ &= \frac{\hat{h}^T(H+\lambda I)^{-1}\hat{h} - \hat{h}^T(H+\lambda I)^{-1}H(H+\lambda I)^{-1}\hat{h}}{\hat{h}^T(H+\lambda I)^{-1}\hat{h}} \\ &= \lambda\frac{\hat{h}^T(H+\lambda I)^{-2}\hat{h}}{\hat{h}^T(H+\lambda I)^{-1}\hat{h}}.\end{aligned} \tag{12}$$

which is controlled by the coaction of samples coming from $p(x)$ and $q(x)$, the kernel width σ and the regularization parameter λ. Equation (12) can be transformed to $(1 - RD)\hat{\Theta}^T\hat{h} = \hat{\Theta}^T H\hat{\Theta}$, which directly shows the relationship (or difference) between the two expressions. In this case, we propose to select a proper λ by controlling RD, so that the difference is neither too large nor too small. The challenge of choosing an unbounded $\lambda > 0$ turns into choosing a range-constrained $0 < RD < 1$. In particular, the designer should explore a set of values λ during the training phase, and choose the largest value whose corresponding RD is smaller than a predefined constant RD_0.

III. Threshold Setting

In the stationary case, the unknown distribution of $\hat{D}^2$ is associated with the comparison of two independent subsets containing i.i.d. samples drawn from distribution $p(x)$ and $q(x) = p(x)$. Such distribution is expected to change once distribution $q(x)$ differs from distribution $p(x)$ due to concept drift. $\hat{D}^2$ values are, as we should expect, sensitive to both the cardinality of n and $p(x)$.

In this paper, we are addressing the case where data come continuously, in data streams. As such, the two subsets containing information about distributions $p(x)$ and $q(x)$ are windows Z_p and Z_q opened over the data streams. The problem we should ask is "when and at which level of confidence" data coming from Z_q are no more coherent with those coming from Z_p, i.e., concept drift occurred in Z_q.

Given the limited data set of size N_t provided for training and the fact that n data are requested to generate an LSDD value, we present a bootstrap mechanism to generate enough LSDD values from N_t to be able to infer their distribution. It must be outlined that we are not providing the distribution of the LSDD values generated with infinite training samples and independent windows but, instead, the distribution of LSDD

values bounded by the fact we consider an n data window and the bootstrap mechanism. Clearly, if N_t is large enough, we can consider independent windows in order to estimate LSDD values. However, the methodology does not change. Moreover, it should be pointed out that having a small N_t would allow us to become immediately operational after the method has been configured and that a small n permits the method to keep under control its computational complexity. The effectiveness of using bootstrap on similar problems has been validated by previous studies [34]–[36].

De Brabanter *et al.* [34] derived thresholds for change detection so as to satisfy some quantile levels of the bootstrapped-based distribution. Sugiyama *et al.* [36] also approximated the p-value of measured Pearson divergences as thresholds for the density-ratio estimation. Burghouts *et al.* [37] proved that the L^p-norm of two nonidentical distributed and correlated bounded feature vectors satisfies a Weibull distribution. Khan *et al.* [38] modeled the L^2-norm dissimilarity to identify the singularities in dikes with a mixture of gamma and uniform distributions where the uniform distribution accounts for some possible anomalies. Unfortunately, no direct results indicate which distribution should be preferred to fit the L^2-norm dissimilarity of two densities. In this paper, we derive a threshold as in [34] and [36] by estimating the $1-\mu$ quantile. As recommended in [7], the selected threshold ensures the controllability of FP rate μ

$$\Pr(\hat{D}^2 > T_\mu) = \mu \tag{13}$$

and the corresponding value of ARL0 is $1/\mu$. Thus, given an acceptable FP rate or the ARL0 value, the threshold T is derived by (13).

IV. LSDD-CDT

An LSDD-CDT can now be designed by exploiting the information content provided by two windows. As mentioned earlier, the first window Z_p includes samples extracted according to reference distribution $p(x)$. The second one Z_q is assumed to contain data generated according to distribution $q(x)$, different from $p(x)$ only when data contain a change in stationarity. This latter window slides in time so that new samples are hosted as they arrive and oldest ones are discarded.

We present a reservoir sampling procedure [39] to manage the data in the reference window Z_p with both old and new samples, which makes the new set Z_p more sensitive to changes as validated in [27]. It permits Z_p to be updated by integrating some instances not present in the training set, which allows the detection mechanism to become immediately operational after its configuration without requiring a huge training data set. Its effect is positive provided that data do not undergo any change in stationarity until the insertion probability goes to zero. Then, $\hat{D}^2$ values are produced on the updated Z_p and Z_q sets, and compared with the threshold T associated with predefined FP rate μ according to (13).

A. Three-Level Threshold Mechanism

We propose a three-level threshold mechanism to be more sensitive to changes (i.e., keep low FNs), yet maintaining the same FP rate.

Algorithm 1 LSDD-CDT

Input: Training set with N_t samples, window size $n < N_t/2$, FP rates $\mu_s > \mu_w > \mu_c$, number of bootstraps m;
Output: Either change detected with its location or no change.
1: Compute the LSDD values (11) from the training set with bootstrap; determine parameters σ and λ;
2: Derive three thresholds T_S, T_W, T_C from the estimated LSDD values $\hat{D}^2$ according to the predefined FP rates μ_s, μ_w, μ_c on (13);
3: Prepare the reference Z_p and testing Z_q windows; $i = 1$;
4: **while** (1) **do**
5: Compute estimate $\hat{D}$ by comparing Z_p and Z_q according to (11);
6: **if** $\hat{D} > T_W$ or in *warning* state **then**
7: Set/keep the warning alarm; Stop updating Z_p;
8: **if** $\hat{D} > T_C$ **then**
9: Change detected at *warning* point P_W;
10: Store the change confirmation point P_C;
11: *Break / reaction*;
12: **end if**
13: **if** $\hat{D} < T_S$ or in *warning* state for n samples **then**
14: Clear the warning alarm;
15: Update Z_p with reservoir sampling.
16: **end if**
17: **else**
18: Update Z_p with the reservoir sampling;
19: Update Z_q with the sliding strategy.
20: $i = i + 1$;
21: **end if**
22: **end while**

The proposed approach encompasses a safe threshold T_S, a warning threshold T_W, and a change threshold T_C. The thresholds in order T_S, T_W, and T_C are associated with decreasing FP rates according to (13) to clear a warning, provide a warning, and confirm the detected change, respectively. In this case, a potential change is soon detected and extra information is then accounted for assessing its nature. When a $\hat{D}^2$ value exceeds T_W associated with a high FP rate μ_w, a *warning* flag is raised in correspondence with the latest sample P_W. Once entered in a potential change detected state, the testing window Z_q slides to collect extra new samples to further assess the nature of the change (either an FP or a true change). If the new computed $\hat{D}^2$ value goes above threshold T_C associated with a low FP rate μ_c, the change is confirmed at point $P_C > P_W$; conversely, if the value is below T_S associated with an FP rate μ_s, or the *warning* state continues even with n new collected samples, the warning alarm is cleared and considered to be a false alarm. When changes are confirmed, we stop testing for changes in stationarity and react since the method perceived a change with probability $1-\mu_c$; if the *warning* is cleared, the two windows continue to slide or update normally. The detailed procedure is described in Algorithm 1, steps 4–22.

The mechanism also permits to detect the change location, i.e., at which sample the change occurred (or was perceived).

More in detail, when a change is detected, the *warning* point P_W corresponding to threshold T_W is considered to be the estimate of the change location $\hat{P} = P_W$. In this case, if reaction to the change mechanisms is needed, we can exploit samples between P_W and P_C that, being associated with the new stationary state can be used to update the application.

B. LSDD-CDT Algorithm

The detailed algorithm for change detection is given in Algorithm 1. Since the aim of this paper is to focus on the change detection problem, aspects related to reactions to the change are not considered (step 11). For the interested reader, an example of the reaction procedure is that associated with a *detect & react* framework, as in just-in-time classifiers [18], [19], where the application reacts to maximize performance once the change is detected.

In applications requesting real-time change detection, memory requirements and computational complexity are aspects to be investigated. It emerges that the memory requested during the operational phase is small, since only $max(2n, N_t)$ samples need to be stored, while the computational complexity of executing a test is $O(n^2)$.

V. Experiments

To validate the effectiveness of the proposed LSDD-CDT, we provide a comprehensive comparison on different applications.

We also introduce two extra tests belonging to this family of CDT and addressing different operational strategies. The first test is an LSDD detection test with two sliding widows (LSDD-Sli); the CDT does not use the reservoir sampling strategy. The second test is based on an ensemble of several reference windows Z_p (LSDD-Ens) during the testing phase, whereas the training step is the same as that in LSDD-CDT. The reference windows adapt with reservoir sampling independently, and are compared with the testing window to detect possible changes. A change is confirmed with the majority voting mechanism. Three well-known methods are also considered for comparison, i.e., the KNN-based test [23], [40], the H-ICI CDT [41], and CPM tests [9], [10].

The KNN-based test aims at monitoring statistics $T_{k,n}$, which indicates how close the two distributions are. For a fair comparison, we use the same training and detection strategy to calculate thresholds and detect changes; k is set to 3 to obtain an FN rate aligned with that of the proposed method.

The H-ICI CDT is a two-level hierarchical CDT whose first level uses the ICI CDT [21], and the second one uses the Hotellings T-square statistic [42]. CPM-LP and CPM-CvM are two general purpose change-point methods suggested in [9] and [10].

Twelve applications are proposed containing both simulated (D1-6 and D12) and real data (D7-11). Simulated data sets are desirable, since the performance of the method can be assessed in various conditions and also the change points can be controlled. In particular, applications D3-11 are well-known benchmarks used to assess detection performance; different concept drifts, such as abrupt, drift, and precision degradation changes, are injected into applications D7-9 and D11; multiple changes are finally considered in D12. More in detail, the following holds.

1) Application D1 generates data drawn from a normal distribution $N(0, 0.5)$ that shifts to $N(0.2\ 0.5)$.
2) Applications D2 refers to a 3-D application, and data satisfy a multivariate normal distribution $N([0, 0, 0], [0.5, 0, 0; 0, 0.5, 0; 0, 0, 0.5])$ that shifts to $N([0, 0, 0], [0.5, 0.4, 0.4; 0.4, 0.5, 0.4; 0.4, 0.4, 0.5])$.
3) Application D3 is a two-class rotating mixture of Gaussians [43] that the class centers shift from $\mu_1 = [1/\sqrt{2}, 1/\sqrt{2}]$, $\mu_2 = [-1/\sqrt{2}, -1/\sqrt{2}]$ to $\mu_1 = [1/\sqrt{2}, -1/\sqrt{2}]$, $\mu_2 = [-1/\sqrt{2}, 1/\sqrt{2}]$. The covariance is fixed as $\Sigma_1 = \Sigma_2 = [0.5, 0; 0, 0.5]$.
4) Application D4 refers to a 2-D circle problem [44], so that data satisfy equation $(x_1 - a)^2 + (x_2 - b)^2 \leq r^2$. Concept drift is associated with changes in the radius r from 0.2 to 0.3. $a = b = 0.5$ and x_1, x_2 are random variables uniformly distributed from interval [0, 1].
5) Application D5 refers to the SineV problem [44], where data satisfy $x_2 \leq a\sin(bx_1 + c) + d$, and changes affect parameter d with values shifting from -5 to 4. We selected $a = b = 1, c = 0$, and x_1 and x_2 are random variables uniformly distributed from intervals [0, 10] and $[-10, 10]$, respectively.
6) Application D6 refers to a moving hyperplane [44], so that $x_{d+1} \leq -a_0 + \sum_{i=1}^{d} a_i x_i$; concept drift is induced by operating on a_0 with values moving from -1 to -3.2. $a_1 = a_2 = 0.1$, and x_1 and x_2 are random variables uniformly distributed from interval [0, 1] and x_3 from interval [0, 5].
7) Applications D7-9 [45] refer to a hairdryer application present in the MATLAB toolkit. Since an ARMAX model well describes the monodimensional input-output process, changes are detected by inspecting the residuals $e(t) = \hat{z}_2 - z_2$ between the output $\hat{z}_2$ as provided by the ARMAX model and the real measurement z_2. Three types of concept drifts are considered according to the additive model, $\hat{z}_2(t) = z_2(t) + \delta(t)$, $\delta(t)$ being the perturbation added at sample 501 and defined as follows.
   a) *Application D7:* $\delta(t) = 0.1\text{mean}(z_2)$ represents an abrupt perturbation affecting z_2.
   b) *Application D8:* A linear drift perturbation $\hat{z}_2(t) = z_2(t) + \delta(t) \cdot t$, so that when $t = 1000$, $\delta(t) \cdot t = 0.1\text{mean}(z_2)$.
   c) *Application D9:* A precision degradation case modeled as $\hat{z}_2(t) = z_2(t) + \delta(t)$ with $\delta(t)$ randomly drawn from distribution N(0, 0.1).
8) Data of application D10 contain 4000 samples collected from a combined cycle power plant [46], [47]. The four features consist of hourly average ambient variables, i.e., temperature, ambient pressure, relative humidity, and exhaust vacuum, and are used to predict the net hourly electrical energy output. We normalize the data set into interval [1, 1]. Changes are injected at sample 2001 with the normalized temperature shifting from x_1 to $-x_1$.
9) Application D11 refers to data acquired from a monitoring system deployed on the Alps to monitor a potential

rock collapse [48]. A total of 11 520 samples, acquired over 80 days with 10 min sampling time, refer to three temperature sensors. The structure of the experiment is in line with applications D7-9, with changes affecting a single sensor only from sample 9000, as shown in Fig. 3(a)–(c). Since these temperatures are correlated, we model sensor interdependence with an ARMAX model receiving two sensor data streams as inputs and the third one as output. Three ARMAX models are identified, each of which generating a residual to be inspected for change detection. Since the residuals introduce aperiodicity associated with the day–night evolutions, we detect changes by inspecting nonoverlapped windows of one-day samples.

10) Application D12 refers to four scenarios each of which containing multiple changes (please refer to Fig. 4). Data in stationarity conditions satisfy a unidimensional normal distribution $N(0, 0.1)$, and the added changes are as follows.
    a) *Case 1:* Two different brief-in-duration changes occur successively. The first one (1#) follows distribution $N(0, 0.25)$, and the second one (2#) $N(0.25, 0.1)$. Each concept drift consists of 50 samples, and changes are separated by 50 stationary samples.
    b) *Case 2:* One concept drift of 50 samples occurs and satisfies distribution $N(0.25, 0.1)$.
    c) *Case 3:* Two different changes occur as with Case 1. Here, each drift lasts 100 samples, and is separated by 50 stationary samples.
    d) *Case 4:* A gradual drift occurs with both mean and standard variance. The new distribution is $N(0.1\sin((i\pi)/(200)), 0.1 + (0.15i)/(1000))$, $i = 1, \ldots, 1000$.

Other setting parameters are defined based on our experience as follows. Since we are considering a scenario with limited training set, the training set N_t is composed of 400 samples, and the number of bootstraps m is 2000. The FP rates μ_s, μ_w, and μ_c corresponding to T_S, T_W, and T_C, respectively, are set to 2%, 1%, and 0.1%, i.e., the corresponding values of ARL0 are 50, 100, and 1000 samples. It is worth noting that we just make a fair comparison between our methods and others under the same condition, e.g., the same FP rates. RD_0 is set to 0.25 by experience, and the optional values of λ are generated by the MATLAB function logspace $(-2, 1, 20)$ that includes 20 values between 0.01 and 10. The number of reference windows in LSDD-Ens is 5. A total of 500 experiments are repeated for each application to assess results.

Above-mentioned settings are appropriate for LSDD-Sli, LSDD-Ens, LSDD-CDT, and KNN-based tests, since they follow the same detection procedure. Differently, the H-ICI CDT is configured as proposed in [41]. The ARL0 values of CPMs are set to 1000, i.e., the predefined FP rate is 0.1% in line with μ_c.

We consider the FP and FN rates, as well as the detection delay and computational time (CT) as performance indexes.

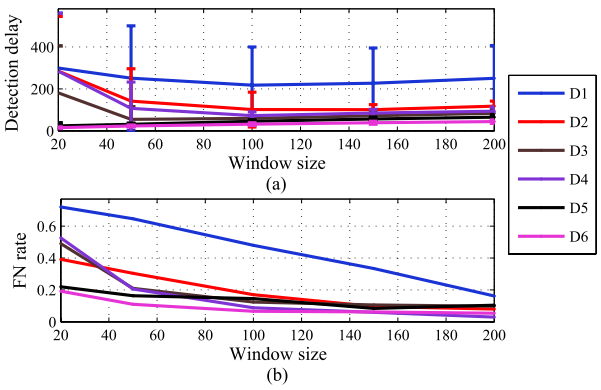


Fig. 1. Influence of window size n on (a) detection delay and (b) FN rate.

TABLE I
TESTING THE REAL FP RATE

	predefined FP rate μ_c				
	5e-2	1e-2	5e-3	2e-3	1e-3
D1	4.88e-2 (2.31e-2)	1.07e-2 (1.08e-2)	5e-3 (7.1e-3)	2.5e-3 (5.2e-3)	1.1e-3 (3.5e-3)
D2	5.21e-2 (2.21e-2)	1.04e-2 (1.11e-2)	5.8e-3 (7.8e-3)	2.4e-3 (4.9e-3)	1.5e-3 (3.9e-3)
D3	4.96e-2 2.25e-2	1.03e-2 1.02e-2	5.5e-3 7.4e-3	2.5e-3 5.1e-3	1.2e-3 3.8e-3
D4	4.86e-2 (2.11e-2)	1e-2 (1.04e-2)	4.7e-3 (7.2e-3)	2.5e-3 (5.2e-3)	1.2e-3 (3.6e-3)
D5	5.22e-2 (2.39e-2)	9.9e-3 (1.04e-2)	5.5e-3 (7.8e-3)	2.4e-3 (5.1e-3)	1.2e-3 (3.7e-3)
D6	5.08e-2 (2.32e-2)	1.01e-2 (1.07e-2)	4.8e-3 (7.2e-3)	2.3e-3 (4.9e-3)	1.1e-3 (3.3e-3)

1) *FP Rate [FP(%)]:* It represents the percentage of experiments where a test erroneously detects a change.
2) *FN Rate [FN(%)]:* It represents the percentage of experiments where the existing change is not detected.
3) *Delay (in Samples):* It measures the promptness in detection latency by considering the detection delay. We record a delay only when the real change is detected; both the mean and the standard deviation of the delay values are computed.
4) *CT (CT in Seconds):* It measures the execution time needed to perform the tests, including the training phase (reference platform: ThinkCentre M4300t Intel i5 core running at 3.1 GHz, 4G RAM). Both the means and the standard deviations are computed.

Although the estimated change location $\hat{P}$ of LSDD-CDT (in Algorithm 1, step 10) is known, we take the change confirmation point P_C as the change detected location to calculate detection delay for fair comparison in our experiments.

A. Influence of the Window Size

In order to investigate the impact of the window size, we consider $n \in \{20, 50, 100, 150, 200\}$ for performance comparison (for large n, the window size gets too close to the training size with a degradation in performance). Results on the simulated applications D1-6, averaged over 500 trials, are shown in Fig. 1 with a $\pm$ standard deviation.

TABLE II
ASSESSING CHANGE DETECTION PERFORMANCE ON DIFFERENT APPLICATIONS

		LSDD-Sli	LSDD-Ens	LSDD-CDT	KNN	H-ICI	CPM-LP	CPM-CvM
D1	FP(%)	14	2.4	13	50	0	61.4	62.8
	FN(%)	39.6	21.2	20.2	13.8	36.6	0	0
	Delay(sample)	108.53(69.3)	215.13(180.34)	199.17(182.64)	280.1(246.61)	566.06(231.2)	79.95(51.16)	64.35(40.58)
	CT(s)	16.13(2.46)	32.2.69(8.19)	15.66(2.14)	15.88(2.78)	0.043(0.0099)	-	-
D2	FP(%)	18	2.6	12.2	46.4	0	64.6	62.2
	FN(%)	7.2	2	2.4	0	100	12.4	15
	Delay(sample)	137.75(151.08)	116.92(98.66)	120.63(105.61)	56.72(16.59)	*ND*	389.82(275.06)	423.89(305.58)
	CT(s)	14.41(2.48)	37.38(5.6)	14.16(0.89)	14.8(1.42)	0.026(0.021)	-	-
D3	FP(%)	14	2.6	11.6	33.6	0	92.8	99.2
	FN(%)	5.8	4	4	0	100	0.2	0
	Delay(sample)	109.9(81.37)	94.29(37.91)	100.27(70.12)	62.29(18.89)	*ND*	244.28(390.16)	378.99(393.82)
	CT(s)	13.62(1.91)	31.81(5.26)	13.6(1.8)	14.68(1.29)	0.02(0.005)	-	-
D4	FP(%)	11.6	2.2	10.6	46.2	0.2	13	54.8
	FN(%)	0	0	0	0	0	77.4	22
	Delay(sample)	73.13(16.14)	70.01(12.49)	69.3(17.58)	58.91(15.58)	211.08(30.13)	118.81(219.88)	301.35(310.77)
	CT(s)	14.36(1.29)	25.65(2.12)	14.37(1.28)	17.76(1.58)	0.033(0.024)	-	-
D5	FP(%)	15.2	3.6	12.4	42.8	0	7.8	0
	FN(%)	0	0	0	0	0	82.4	100
	Delay(sample)	39.48(7.91)	37.32(6.66)	36.81(7.43)	41.38(13.57)	137.8(7.88)	119.63(22.24)	*ND*
	CT(s)	13.29(1.57)	29.66(2.73)	13.32(1.49)	17.78(1.53)	0.029(0.0012)	-	-
D6	FP(%)	14	3.2	13.8	42.2	0	64.8	67.2
	FN(%)	0	0	0	0	0	12.6	10
	Delay(sample)	26.17(5.83)	24.8(5.18)	24.64(5.68)	38.89(12.83)	127.72(12.23)	257.04(298.14)	286.74(293.5)
	CT(s)	12.59(1.57)	25.21(2.63)	12.54(1.44)	18.5(1.96)	0.031(0.0022)	-	-
D7	FP(%)	12.2	1.4	15.6	15.6	0	100	0
	FN(%)	0	0	0	0	0	0	0
	Delay(sample)	23.3(1.61)	21.1(3.99)	21.9(7.23)	46.25(17.23)	160(0)	*ND*	6(0)
	CT(s)	12.39(1.19)	14.53(1.34)	12.38(1.17)	13.64(0.7)	0.013(0.0007)	-	-
D8	FP(%)	10.2	3	13.2	15.4	0	100	0
	FN(%)	0	0	0	0.4	0	0	0
	Delay(sample)	214.78(10.96)	193.82(51.8)	192.3(76.37)	166.59(88.71)	203.92(31.14)	*ND*	58(0)
	CT(s)	12.92(1.07)	17.76(1.95)	12.82(1.09)	14.23(0.99)	0.017(0.0012)	-	-
D9	FP(%)	13.4	3.2	13.6	15	0	100	0
	FN(%)	0	0	0	0	0	0	0
	Delay(sample)	48.4(5.34)	47.57(8.16)	46.64(10.3)	47.09(17.52)	126.16(6.97)	*ND*	36.74(10.75)
	CT(s)	12.17(0.93)	14.8(1.09)	12.16(0.93)	13.64(0.68)	0.012(0.0007)	-	-
D10	FP(%)	0.8	0	0.8	68.4	0	100	100
	FN(%)	0	0	0	0	100	0	0
	Delay(sample)	74.38(7.4)	88.27(4.86)	84.85(15.31)	40.04(13.99)	*ND*	*ND*	*ND*
	CT(s)	63.45(4.62)	133.83(6.77)	63.59(4.29)	55.96(8.28)	0.053(0.0052)	-	-

As expected, the FN rate decreases as the window size enlarges while the detection delay is application-dependent and weakly affected. Fig. 1(a) indicates that most of applications show a good detection promptness when n is around 100, and we consider this window size in the sequel.

B. Influence of a Predefined FP Rate

In order to validate how the real FP rates of our method are consistent with the predefined FP rate μ_c, we conducted some experiments on stationary applications D1-6.

Experiments are designed as follows. The predefined FP rates range in the {5e-2, 1e-2, 5e-3, 2e-3, 1e-3} set. For a given predefined FP rate μ_c of each application, 100 tests, as one trial, are carried out to measure the real FP rate; then, 500 trials are executed to compute the mean and the standard deviation of the FP rate. Experimental results are shown in Table I. The real FP rates are close to the predefined ones, which means that the proposed method is reasonably effective in controlling the FP rates.

Another experiment shows how the predefined FP rates, i.e., the corresponding thresholds, affect the detection latency and the FN rate. The set of considered predefined FP rate is $\mu_c \in$ {5e-3, 2e-3, 1e-3, 5e-4, 1e-4}.

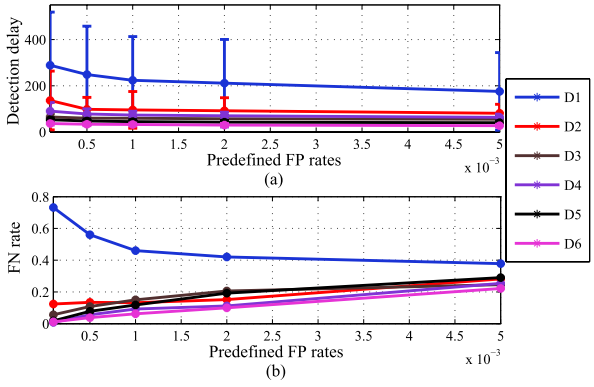


Fig. 2. Influence of predefined FP rate on (a) detection delay and (b) FN rate.

Results with their mean and standard deviations are shown in Fig. 2. As expected, with the increase of the FP rate, the detection delay on all these applications slightly decreases. The FN rate in D1 decreases as per the intuition, whereas the

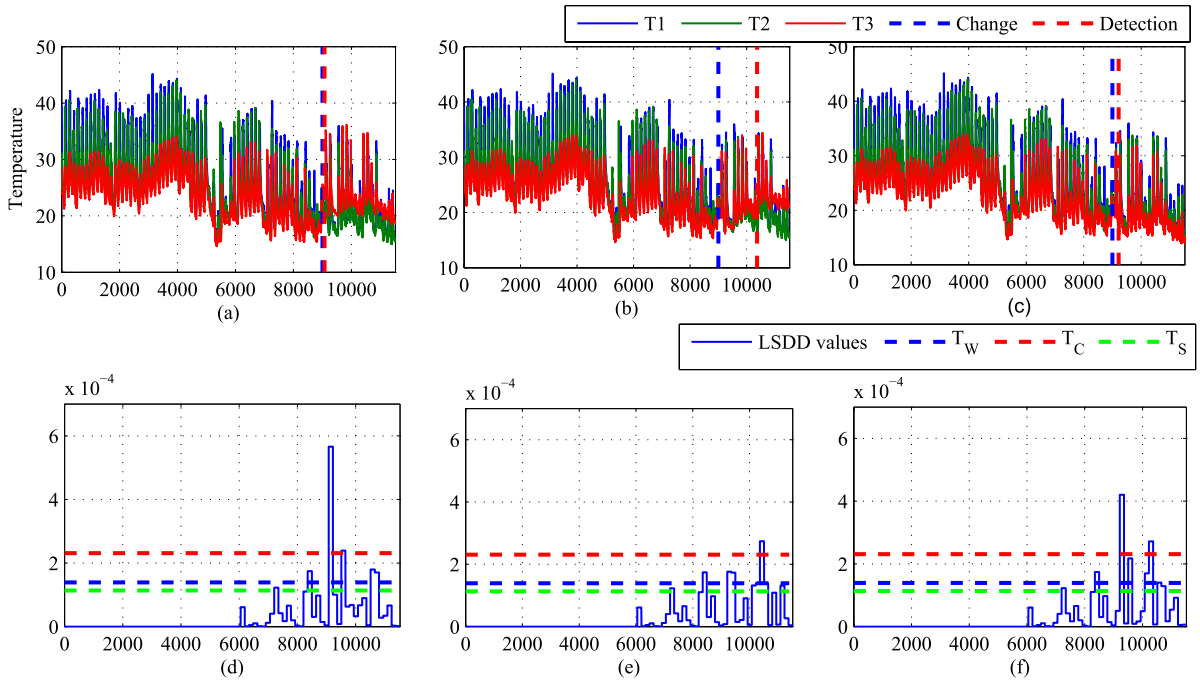


Fig. 3. Experiment results on D11 with three types of changes. (a) Temperatures with an abrupt perturbation affecting T3. (b) Temperatures with a linear drift perturbation affecting T3. (c) Temperatures with a precision degradation affecting T3. (d) Estimated LSDD values with changes shown in (a) and the derived thresholds. (e) Estimated LSDD values with changes shown in (b) and the derived thresholds. (f) Estimated LSDD values with changes shown in (c) and the derived thresholds. T1–3 refer to temperatures gathered from three different sensors; Change: change location; Detection: location where we detect changes; LSDD values: estimated LSDD values; Tw: warning threshold; Tc: change threshold; Ts: safe threshold.

FN rates in D2-6 show a slow rising trend. We comment that FP rate $\mu_c = 0.1\%$ appears to be appropriate to obtain small detection delays as well as small FN rates.

C. Assessing Change Detection Performance

This section aims at comparing the performance among the family of LSDD-CDTs and other competing methods. The window size n is set to 100 and $N_t = 400$, but in D10 given the high dimensionality of inputs, we set $n = 200$ and $N_t = 1000$. The first five methods are implemented in the MATLAB, while CPMs operate using the R package cpm. In this case, given the unfair time comparison on different platforms, we do not record the execution time of CPMs. The comparative analysis of detection performance is shown in Table II where *ND* represents *not detected*.

H-ICI has the smallest FP and FN rates and the smallest CT in most applications. The reason is that it segments observations into nonoverlapped windows, which saves computation. However, the method fails to detect changes in D2-3 and D10 as expected, because H-ICI reduces multidimensional applications to several monodimensional cases at first and then detects changes in each dimension. Moreover, the detection delay is much higher than LSDD-CDT in all these applications.

CPMs do well in D1 with the shortest detection delay but with high FP rates. The performance of accuracy and promptness in D2-6 is almost the worst either on the FP or the FN rates, and the detection comes with a significant latency. Moreover, CPM-LP behaves badly in applications D7-10, since it always detects an FP.

KNN-based test shows a similar performance with LSDD-CDT in promptness and CT. However, it has much higher FP rates, and always reports a false detection before the real change happens.

The comparison between LSDD-Sli and LSDD-CDT shows that the latter introduces both lower FP and FN rates. This happens mostly in applications D1-6, which indicates the effectiveness of adopting the reservoir sampling. As we expected, LSDD-Ens works better than LSDD-CDT in detection accuracy and promptness, with both lower FP and FN rates. However, it costs more time to compute the density differences, since we use an ensemble with five reference windows.

Based on the analysis mentioned earlier, we achieve the following conclusions. H-ICI shows an obvious advantage in accuracy and CT, but with a higher delay, and it fails to detect changes in truly multidimensional applications, e.g., D2-3 and D10; CPMs do well in detection promptness on data sets from normal distribution, whereas they almost fail to detect changes in other cases with either too high FP or FN rates; LSDD-Ens works well in all the applications with a slightly higher computational cost compared with other methods. Under some specific occasions where only one or two indexes are required, such as high accuracy or low CT, each of these methods would show a desired performance. For instance, we recommend the ensemble procedure of LSDD-CDT when execution time is not a strong constraint. It is worth stressing that all LSDD-CDTs permit to set the FP rates at a desired level, and do a good job in detection promptness under acceptable FN rates in all these applications, that is, the LSDD-CDT family of tests provides an excellent integrated and consistent performance. In real applications, methods with a controllable FP rate and small detection delay are always preferred. All-in-all LSDD-Ens is performing very

TABLE III
ASSESSING THE DETECTION PERFORMANCE WITH MULTIPLE CHANGES

n		Case 1		Case 2	Case 3		Case 4
		1#	2#		1#	2#	
100	FP(%)	8.4%		6.6%	5%		4.6%
	DetR(%)	36.2%	55.4%	93.4%	95%	0	95.4%
	FN(%)	0		0	0		0
	Delay(s)	56.56(19.54)	19.73(9.65)	29.15(6.47)	61.58(14.81)	*ND*	97.55(34.25)
200	FP(%)	4.8%		7.2%	3.2%		2.6%
	DetR(%)	7.8%	87.4%	92.8%	78.8%	18%	97.4%
	FN(%)	0		0	0		0
	Delay(s)	50.69(19.16)	21.95(8.83)	48.72(40.34)	83.52(20.25)	11.04(6.07)	123.57(35.74)

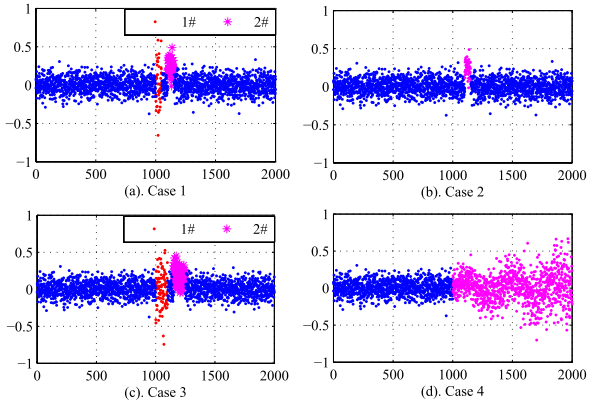


Fig. 4. Four cases with multiple changes. (a) Case 1. (b) Case 2. (c) Case 3. (d) Case 4.

nicely on most of the indexes and is particularly effective on truly multidimensional applications where inspection of a single dimension does not suffice (situation where the H-ICT fails).

The next experiment refers to real application D11 where different types of changes are added to the temperature sensor $T3$. The first 5740 samples, associated with 40 days, constitute the training set, and an abrupt, a linear drift, and a precision degradation concept drift are added from sample 9000, as shown in Fig. 3(a)–(c), respectively. The change location index is shown with a blue dashed-dotted line, and the detection index in red dashed-dotted line. All these changes can be detected immediately once enough samples are collected to show the significance of changes and this explains the large detection delay introduced by the linear drift. The LSDD values are shown in Fig. 3(d)–(f), as well as the three thresholds.

The last experiment focuses on the performance in detecting multiple changes in data streams. Here, we introduce a new index "detection rate (DetR)" to record the rates (over 500 trials) that the current change is detected timely before the next one occurs. In spite of using the default window size $n = 100$, which behaves properly in our applications, we also consider a larger size $n = 200$. The larger size can include several changes. The four cases are shown in Fig. 4, and the detection performance with different window sizes is recorded in Table III. The comparative analysis holds the following.

1) *Case 1 Versus Case 2:* The first changes (1#) in Case 1 are mostly undetected, since the few available nonstationary samples are not enough to guarantee a statistically significant difference (change). However, when the testing window slides to contain more samples associated with the changes (Case 1), the method detects "an equivalent" change faster than in Case 2. That is, the combination of changes 1# and 2# introduces an "integral" effect.
2) *Case 1 Versus Case 3:* In the latter case, there are more nonstationary samples, and the first changes (1#) are detected more accurately.
3) *Case 4:* It models a gradual concept drift scenario where the pdf changes after each sample. When $q(x)$ is different significantly from that of $p(x)$, changes are detected.
4) $n = 100$ *Versus* $n = 200$*:* In Cases 1 and 3, the nonstationary samples are few, so that larger windows ($n = 200$) may include more stationary samples, which weaken the difference between $q(x)$ and $p(x)$. Therefore, the tests become less sensitive to the first changes (1#) in both cases with lower DetRs.

VI. CONCLUSION

In this paper, we propose a novel pdf-free CDT to monitor data streams, which is based on the LSDD method. The method can deal with multidimensional applications whose pdfs are continuous without knowing any priors. By using bootstrap we induce procedure, the distribution of derived LSDD values $\hat{D}^2$ is constructed in stationary conditions so that thresholds needed to claim a change are obtained with predefined FP rates. A three-level threshold mechanism is proposed to be sensitive to small changes (and hence keep under control the FP rate); at the same time, it offers a way to accurately identify change location. Experiments show that the proposed LSDD-CDT works well both in terms of promptness and accuracy in all considered applications. We also provide a version of LSDD-CDT by using an ensemble of several reference windows; LSDD-Ens shows to be particularly effective in true high dimensional applications at a high computational cost.

REFERENCES

[1] R. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *IEEE Trans. Neural Netw.*, vol. 22, no. 10, pp. 1517–1531, Oct. 2011.
[2] D. Brzezinski and J. Stefanowski, "Reacting to different types of concept drift: The accuracy updated ensemble algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 1, pp. 81–94, Jan. 2014.

[3] D. S. Pereira Salazar, P. J. Leitao Adeodato, and A. Lucena Arnaud, "Continuous dynamical combination of short and long-term forecasts for nonstationary time series," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 1, pp. 241–246, Jan. 2014.

[4] C. Alippi and M. Roveri, "An adaptive CUSUM-based test for signal change detection," in *Proc. Int. Symp. Circuits Syst.*, May 2006, pp. 5752–5755.

[5] C. Alippi, G. Boracchi, and M. Roveri, "Change detection tests using the ICI rule," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul. 2010, pp. 1–7.

[6] C. Alippi, L. Bu, and D. Zhao, "A prior-free encode-decode change detection test to inspect datastreams for concept drift," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Aug. 2013, pp. 1–6.

[7] D. M. Hawkins and K. D. Zamba, "Statistical process control for shifts in mean or variance using a changepoint formulation," *Technometrics*, vol. 47, no. 2, pp. 164–173, 2005.

[8] K. D. Zamba and D. M. Hawkins, "A multivariate change-point model for statistical process control," *Technometrics*, vol. 48, no. 4, pp. 539–549, 2006.

[9] 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.

[10] G. J. Ross and N. M. Adams, "Two nonparametric control charts for detecting arbitrary distribution changes," *J. Quality Technol.*, vol. 44, no. 2, pp. 102–116, 2012.

[11] D. M. Hawkins, P. Qiu, and C. W. Kang, "The changepoint model for statistical process control," *J. Quality Technol.*, vol. 35, no. 4, pp. 355–366, 2003.

[12] A. M. Mood, "On the asymptotic efficiency of certain nonparametric two-sample tests," *Ann. Math. Statist.*, vol. 25, no. 3, pp. 514–522, 1954.

[13] Y. V. E. S. Lepage, "A combination of Wilcoxon's and Ansari-Bradley's statistics," *Biometrika*, vol. 58, no. 1, pp. 213–217, 1971.

[14] F. J. Massey, Jr., "The Kolmogorov-smirnov test for goodness of fit," *J. Amer. Statist. Assoc.*, vol. 46, no. 253, pp. 68–78, 1951.

[15] T. W. Anderson, "On the distribution of the two-sample Cramér–Von Mises criterion," *Ann. Math. Statist.*, vol. 33, no. 3, pp. 1148–1159, 1962.

[16] H. Raza, G. Prasad, and Y. Li, "Dataset shift detection in non-stationary environments using EWMA charts," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, Oct. 2013, pp. 3151–3156.

[17] H. Raza, G. Prasad, and Y. Li, "Adaptive learning with covariate shift-detection for non-stationary environments," in *Proc. UKCI*, Sep. 2014, pp. 1–8.

[18] C. Alippi and M. Roveri, "Just-in-time adaptive classifiers—Part I: Detecting nonstationary changes," *IEEE Trans. Neural Netw.*, vol. 19, no. 7, pp. 1145–1153, Jul. 2008.

[19] C. Alippi and M. Roveri, "Just-in-time adaptive classifiers—Part II: Designing the classifier," *IEEE Trans. Neural Netw.*, vol. 19, no. 12, pp. 2053–2064, Dec. 2008.

[20] C. Alippi, D. Liu, D. Zhao, and L. Bu, "Detecting and reacting to changes in sensing units: The active classifier case," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 44, no. 3, pp. 353–362, Mar. 2014.

[21] C. Alippi, G. Boracchi, and M. Roveri, "A just-in-time adaptive classification system based on the intersection of confidence intervals rule," *Neural Netw.*, vol. 24, no. 8, pp. 791–800, 2011.

[22] L. I. Kuncheva and W. J. Faithfull, "PCA feature extraction for change detection in multidimensional unlabeled data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 1, pp. 69–80, Jan. 2014.

[23] M. F. Schilling, "Multivariate two-sample tests based on nearest neighbors," *J. Amer. Statist. Assoc.*, vol. 81, no. 395, pp. 799–806, 1986.

[24] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in nonstationary environments: A survey," *IEEE Comput. Intell. Mag.*, vol. 10, no. 4, pp. 12–25, Nov. 2015.

[25] G. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognit. Lett.*, vol. 33, pp. 191–198, Jan. 2012.

[26] S. Sakthithasan, R. Pears, and Y. S. Koh, "One pass concept change detection for data streams," in *Advances in Knowledge Discovery and Data Mining*. Berlin, Germany: Springer, 2013, pp. 461–472.

[27] R. Pears, S. Sakthithasan, and Y. S. Koh, "Detecting concept change in dynamic data streams," *Mach. Learn.*, vol. 97, no. 3, pp. 259–293, Dec. 2014.

[28] T. Dasu, S. Krishnan, S. Venkatasubramanian, and K. Yi, "An information-theoretic approach to detecting changes in multi-dimensional data streams," in *Proc. Symp. Interface Statist. Comput. Sci. Appl.*, 2006, pp. 1–24.

[29] S. Liu, M. Yamada, N. Collier, and M. Sugiyama, "Change-point detection in time-series data by relative density-ratio estimation," *Neural Netw.*, vol. 43, pp. 72–83, Jul. 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608013000270

[30] S. Liu, J. A. Quinn, M. U. Gutmann, T. Suzuki, and M. Sugiyama, "Direct learning of sparse changes in Markov networks by density ratio estimation," *Neural Comput.*, vol. 26, no. 6, pp. 1169–1197, 2014.

[31] M. Sugiyama, T. Kanamori, T. Suzuki, M. C. Du Plessis, S. Liu, and I. Takeuchi, "Density-difference estimation," *Neural Comput.*, vol. 25, no. 10, pp. 2734–2775, Oct. 2013.

[32] T. D. Nguyen, M. C. Du Plessis, T. Kanamori, and M. Sugiyama, "Constrained least-squares density-difference estimation," *IEICE Trans. Inf. Syst.*, vol. 97, no. 7, pp. 1822–1829, 2014.

[33] A. Gretton *et al.*, "Optimal kernel choice for large-scale two-sample tests," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1205–1213.

[34] K. De Brabanter, S. Sahhaf, P. Karsmakers, J. De Brabanter, J. Suykens, and B. De Moor, "Nonparametric comparison of densities based on statistical bootstrap," in *Proc. ECUMICT*, 2010, pp. 179–190.

[35] X. Song, M. Wu, C. Jermaine, and S. Ranka, "Statistical change detection for multi-dimensional data," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2007, pp. 667–676.

[36] M. Sugiyama, T. Suzuki, Y. Itoh, T. Kanamori, and M. Kimura, "Least-squares two-sample test," *Neural Netw.*, vol. 24, no. 7, pp. 735–751, 2011.

[37] G. Burghouts, A. Smeulders, and J. M. Geusebroek, "The distribution family of similarity distances," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 201–208.

[38] A. A. Khan, V. Vrabie, J. I. Mars, A. Girard, and G. D'Urso, "Automatic monitoring system for singularity detection in dikes by DTS data measurement," *IEEE Trans. Instrum. Meas.*, vol. 59, no. 8, pp. 2167–2175, Aug. 2010.

[39] J. S. Vitter, "Random sampling with a reservoir," *ACM Trans. Math. Softw.*, vol. 11, no. 1, pp. 37–57, 1985.

[40] N. Henze, "A multivariate two-sample test based on the number of nearest neighbor type coincidences," *Ann. Statist.*, vol. 16, no. 2, pp. 772–783, Jun. 1988.

[41] C. Alippi, G. Boracchi, and M. Roveri, "A hierarchical, nonparametric, sequential change-detection test," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Aug. 2011, pp. 2889–2896.

[42] R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1992.

[43] G. Ditzler and R. Polikar, "Hellinger distance based drift detection for nonstationary environments," in *Proc. IEEE Symp. Comput. Intell. Dyn. Uncertain Environ.*, Apr. 2011, pp. 41–48.

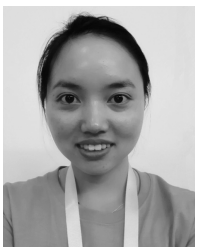
[44] L. L. Minku, A. P. White, and X. Yao, "The impact of diversity on Online ensemble learning in the presence of concept drift," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 5, pp. 730–742, May 2010.

[45] L. Ljung, *System Identification: Theory for the User*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1999.

[46] P. Tüfekci, "Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods," *Int. J. Electr. Power Energy Syst.*, vol. 60, pp. 126–140, Sep. 2014.

[47] H. Kaya, P. Tüfekci, and F. S. Gürgen, "Local and global learning methods for predicting power of a combined gas steam turbine gas steam turbine," in *Proc. Int. Conf. Emerg. Trends Comput. Electron. Eng.*, Dubai, United Arab Emirates, 2012, pp. 13–18.

[48] C. Alippi, R. Camplani, C. Galperti, A. Marullo, and M. Roveri, "A high-frequency sampling monitoring system for environmental and structural applications," *ACM Trans. Sens. Netw.*, vol. 9, no. 4, p. 41, 2013.



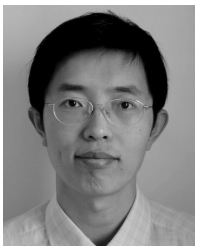
Li Bu (S'15) received the B.S. degree in electronic engineering and automation from the China University of Mining and Technology, Xuzhou, China, in 2012. She is currently pursuing the Ph.D. degree in computer science with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China.

Her current research interests include adaptation and learning in nonstationary environments and computational intelligence.

Cesare Alippi (SM'94–F'06) received the Degree *cum laude* in electronic engineering in 1990, and the Dr.Ing. degree from the Politecnico di Milano, Milano, Italy, in 1995.

He was a Visiting Researcher with University College London, London, U.K., Massachusetts Institute of Technology, Cambridge, MA, USA, ESPCI Paris Tech, Paris, France, Institute of Automation, Chinese Academy of Sciences, Agency for Science Technology and Research, Singapore, and UKobe, Japan. He is currently a Full Professor with the Politecnico di Milano, Milano, Italy, and the Università della Svizzera italiana, Lugano, Switzerland. He holds five patents. He has authored one monograph book, six edited books, and about 200 papers in international journals and conference proceedings. His current research interests include adaptation and learning in nonstationary environments and intelligence for embedded and cyber-physical systems.

Dr. Alippi is a Board of Governors member of the International Neural Network Society, a Board of Directors member of the European Neural Network Society, a Past Vice President education of the IEEE Computational Intelligence Society, a past Associate editor of the *IEEE Computational Intelligence Magazine*, the IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENTS, and the IEEE TRANSACTIONS ON NEURAL NETWORKS. He received the Gabor Award from the International Neural Networks Society and the IEEE Computational Intelligence Society Outstanding Transactions on Neural Networks and Learning Systems Paper Award in 2016, the IBM Faculty Award in 2013, the IEEE Instrumentation and Measurement Society Young Engineer Award, in 2004, and the Knight of the Order of Merit of the Italian Republic in 2011.



Dongbin Zhao (M'06–SM'10) received the Ph.D. degree from the Harbin Institute of Technology, Harbin, China, in 2000.

He was a Post-Doctoral Fellow with Tsinghua University, Beijing, China, from 2000 to 2002. He was an Associate Professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2002. He is currently a Professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, since 2012. He has authored four books, and published over 50 international journal papers. His current research interests include computational intelligence, adaptive dynamic programming, robotics, intelligent transportation systems, and smart grids.

Dr. Zhao is an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS (2012-), the *IEEE Computation Intelligence Magazine* (2014-). He serves as the Chair of Adaptive Dynamic Programming and Reinforcement Learning Technical Committee (2015-), the Multimedia Subcommittee (2015-), and the Travel Grant Subcommittee (2015-) of the IEEE Computational Intelligence Society. He served as a Guest Editor of several international journals. He is involved in organizing several international conferences.