Change Detection in Multivariate Datastreams Controlling False Alarms

Luca Frittoli[1], Diego Carrera[2], and Giacomo Boracchi[1]

[1] DEIB, Politecnico di Milano, via Ponzio 34/5, Milan, Italy
{luca.frittoli, giacomo.boracchi}@polimi.it
[2] STMicroelectronics, via Camillo Olivetti 2, Agrate Brianza, Italy
diego.carrera@st.com

Abstract. We introduce QuantTree Exponentially Weighted Moving Average (QT-EWMA), a novel change-detection algorithm for multivariate datastreams that can operate in a nonparametric and online manner. QT-EWMA can be configured to yield a target Average Run Length (ARL_0), thus controlling the expected time before a false alarm. Control over false alarms has many practical implications and is rarely guaranteed by online change-detection algorithms that can monitor multivariate datastreams whose distribution is unknown. Our experiments, performed on synthetic and real-world datasets, demonstrate that QT-EWMA controls the ARL_0 and the false alarm rate better than state-of-the-art methods operating in similar conditions, achieving comparable detection delays.

Keywords: online change detection, nonparametric monitoring, multivariate datastreams, histograms, false alarms

1 Introduction

Detecting changes in datastreams is a frequently encountered problem in industrial [13] and traffic monitoring [12], security [31], cryptographic attacks [9], and finance [29], to name a few examples. Change detection is also relevant in machine learning, where changes are also known as *concept drifts*, and classifiers have to be adapted as the data-generating process changes [10]. In many of these applications, datastreams have to be processed *online*, i.e., while acquiring observations, and this condition poses crucial, sometimes conflicting, challenges from both algorithmic and technological standpoints. On the one hand, online change-detection algorithms have to analyze virtually unlimited datastreams, while storing a limited amount of data and performing a fixed number of operations per incoming sample. On the other hand, to detect even subtle changes, online change-detection algorithms are requested to perform a sequential analysis, leveraging at time t all the data samples observed until t, in contrast with *one-shot* detectors, which perform independent tests over batches of data.

This paper addresses the crucial problem of controlling false alarms in online change detection, which is particularly important in industrial monitoring and unsupervised drift detection, where each detected change possibly triggers a costly intervention or supervision. Operating at controlled false alarm rates enables allocating the right amount of resources for the supervision, as well as identifying issues in the monitored process when there are more detections than expected. Unfortunately, most online change-detection algorithms that can monitor multivariate datastreams, especially nonparametric ones, fail to control false alarms effectively.

Most change-detection algorithms feature two main ingredients: *i*) a *statistic* with a known response to data following the initial distribution ϕ_0, and *ii*) a *decision rule* to analyze the values of the statistic and report changes. In online monitoring, the statistic T_t considers all the data acquired until time t, and the decision rule typically compares the statistic T_t against a threshold h, which is defined to control false alarms. In offline tests and one-shot detectors analyzing a fixed amount of data, the threshold h is set as a specific quantile of T and computed analytically [23], or by bootstrap [3]. In online monitoring, setting thresholds becomes more complicated, as one typically wants to control the *Average Run Length* (ARL_0), i.e., the average time before raising a false alarm [2]. Controlling the ARL_0 requires defining a sequence of thresholds $\{h_t\}_t$ where each threshold h_t is set as a specific quantile of the statistic T_t conditioned on the fact that the algorithm has not detected any change before t. In both offline and online monitoring, tests based on *nonparametric* statistics are preferable because their thresholds $\{h_t\}_t$ can be set without any information about ϕ_0.

In this paper, we propose *QT-EWMA*, a novel online change-detection algorithm for multivariate datastreams combining a QuantTree histogram (QT) [3] as a general and flexible model for ϕ_0, with an online statistic based on the Exponentially Weighted Moving Average (EWMA) [28]. In particular, we define a novel EWMA statistic to monitor the proportion of incoming samples falling in each bin of the histogram, and use this to define an efficient and practical online change-detection algorithm. The theoretical background of QuantTree guarantees that this test statistic does not depend on ϕ_0 nor the data dimension [3]. Thus we derive an efficient Monte Carlo scheme to compute thresholds controlling ARL_0. By design, these thresholds guarantee a constant false alarm probability over time and, consequently, a fixed false alarm rate at each time instant. Thus, QT-EWMA controls both the ARL_0 and the false alarm rate. Our main contributions are:

- We develop QT-EWMA, an online change-detection algorithm for multivariate datastreams based on a novel statistic that monitors the bin probabilities of a QuantTree histogram (Section 4.1).
- We design an efficient Monte Carlo scheme to compute thresholds controlling the ARL_0 in QT-EWMA, as in [29]. Thanks to the theoretical properties of QuantTree, our thresholds controls both the ARL_0 and the probability of false alarms at a given time, for any distribution ϕ_0 (Section 4.2).
- We propose two simple yet theoretically sound procedures to extend a generic one-shot detector to monitor datastreams while controlling the ARL_0 (Section 5), which we employ as baselines in our experiments.

Our experiments performed on both simulated and real-world datastreams show that QT-EWMA controls the ARL_0 better than the baselines, and Scan-B [22], a competing algorithm based on a *Maximum Mean Discrepancy* (MMD) statistic. Our results also show that QT-EWMA maintains the expected false alarm rate, which Scan-B does not guarantee. Moreover, in the considered real-world datasets, QT-EWMA achieves detection delays on par with online and nonparametric tests implementing powerful statistics based on kernels, which cannot control false alarms. QT-EWMA implementation and thresholds are available for download at `https://github.com/diegocarrera89/quantTree`.

The rest of the paper is organized as follows: in Section 2 we illustrate the literature of online change detection algorithms for multivariate datastreams, while in Section 3 we provide a formal definition of the online change-detection problem. In Section 4 we introduce our proposed solution, together with our procedure to compute the thresholds controlling ARL_0. In Section 5 we illustrate how to extend one-shot change detectors to monitor datastream controlling the ARL_0, and discuss the theoretical guarantees and limitations of these approaches. Section 6 analyzes the computational complexity of QT-EWMA compared to alternative methods and our experiments in Section 7 demonstrate the effectiveness of QT-EWMA on both simulated and real-world datastreams.

2 Related Work

Most change-detection algorithms in the literature are designed to monitor univariate datastreams [13, 29, 28]. The vast majority of these methods cannot be extended to monitor multivariate datastreams, especially those leveraging nonparametric statistics based on ranks [29]. Change detection in multivariate datastreams has often been addressed in multi-stream (multi-channel) settings, i.e., by separately analyzing each input component [25, 32, 8]. However, the hypotheses underpinning the multi-stream monitoring setting are fundamentally different from those of our multivariate datastream change-detection problem. In particular, [25, 32, 8] assume the components of the data points are generated by separate random variables rather than a single multivariate random vector. Moreover, in multi-stream settings, changes typically affect the distribution of a subset of these random variables [32], while, in multivariate settings, more general kinds of distribution changes are admissible [5]. In particular, changes affecting the correlation between components or subtle changes involving the whole vector can be hard to detect by separately analyzing the components.

The first change-detection algorithms for multivariate datastreams assume parametric hypotheses: a remarkable example [33] leverages the Hotelling test statistic [15] to perform online monitoring while controlling the ARL_0. A popular semi-parametric approach consists of reducing the data dimensionality by monitoring the log-likelihood of the observations with respect to a density model fitted on a training set. The most common models for the initial distribution ϕ_0 are Gaussian [12], and Gaussian mixture models [17, 1]. The main limitation of these solutions is the implicit assumption that ϕ_0 can be approximated well by a probability distribution from a known family, which is not guaranteed in general. Other approaches reduce the dimensionality of the data by PCA [18, 27], or by a *strangeness measure* [14, 26]. After reducing the dimensionality, online change detection boils down to monitoring a univariate datastream, for instance by Martingale-based permutation tests [14]. However, none of these methods can be configured before deployment to operate at a target ARL_0.

Kernel methods based on Maximum Mean Discrepancy (MMD) have been introduced for nonparametric one-shot change detection [11]. Recently, the MMD statistic has been employed for online change detection [22, 16], following a sliding-window approach to compare the new observations to a training set. Among these methods, Scan-B [22] is the only where ARL_0 can be set for an unknown distribution ϕ_0. However, the thresholds for this method are defined by analyzing the asymptotic behavior of the ARL_0 when the threshold tends to infinity [22]. As we show in our experiments, this approach does not guarantee an accurate control of the ARL_0 and the false alarm rate. NEWMA [16] also employs a sliding-window monitoring scheme and analyzes the relation between two EWMA statistics with different forgetting factors based on MMD to detect changes online. Unfortunately, thresholds controlling the ARL_0 can only be set when ϕ_0 is known [16], which limits the applicability of this solution.

A very general nonparametric approach consists of modeling ϕ_0 by a histogram [4], as in QuantTree [3], an algorithm to construct histograms over the input domain that are adaptively defined over the distribution ϕ_0 of a training set. A one-shot change-detection method is presented in [3], leveraging a nonparametric statistic to test whether a single batch of data follows ϕ_0. This test cannot be directly used for online change detection. As we show in Section 5, QuantTree and other one-shot detectors can be extended to online change detection controlling false alarms, but these solutions have substantial limitations in maintaining the ARL_0 and in terms of detection power. Another change-detection algorithm based on histograms is presented in [19], and can operate online controlling the ARL_0, but has been tested only on univariate datastreams. An extension to multivariate data is infeasible since the number of bins scales exponentially with the data dimension. Moreover, this algorithm requires to know the analytical expression of ϕ_0, or an accurate approximation, which would require a large training set to be estimated when the data dimension is high [19].

The proposed QT-EWMA overcomes all these limitations, being an online change-detection algorithm controlling the ARL_0 in any practical condition, without requiring to know ϕ_0.

3 Problem Formulation

We address the online change-detection problem in a virtually unlimited multivariate datastream $x_1, x_2 \ldots \in \mathbb{R}^d$. We assume that, as long as there are no changes, all the data samples are i.i.d. realizations of a random variable having unknown distribution ϕ_0. We define the *change point* τ as the unknown time instant when a change $\phi_0 \to \phi_1$ takes place:

$$x_t \sim \begin{cases} \phi_0 & \text{if } t < \tau \\ \phi_1 & \text{if } t \geq \tau \end{cases} . \tag{1}$$

We assume that both ϕ_0 and $\phi_1 \neq \phi_0$ are unknown, and that a training set TR containing N realizations of ϕ_0 is provided.

Online change-detection algorithms assess, for each new incoming sample x_t, whether the sequence $\{x_1, \dots x_t\}$ contains a change point. Typically, a statistic T_t is computed at each incoming x_t, then a decision rule is applied. Usually, the rule consists in controlling whether $T_t > h_t$ for an appropriate threshold h_t, and the detection time t^* is defined as the first time index when this happens:

$$t^* = \min\{t : T_t > h_t\}. \tag{2}$$

The detection time t^* is the first time instant when there is enough statistical evidence to claim that the datastream $\{x_1, \dots x_t\}$ contains a change point.

A fundamental issue in change detection is to define a sequence of thresholds $\{h_t\}_t$ to control false alarms. We measure false alarms by the Average Run Length [2], defined as $\text{ARL}_0 = \mathbb{E}[t^*]$, where the expectation is taken assuming that the whole datastream is drawn from ϕ_0. Thus, the ARL_0 is the average time before a false alarm. Ideally, the ARL_0 of an online change-detection method should be set *a priori*, similarly to Type I error probability in hypothesis testing.

4 Proposed Solution

Here we introduce our novel algorithm QT-EWMA (Section 4.1) and describe the procedure to define its thresholds controlling the ARL_0 (Section 4.2).

4.1 QuantTree Exponentially Weighted Moving Average

The QT-EWMA algorithm (Algorithm 1) leverages a novel online statistic T_t defined over a QuantTree histogram, which is constructed from the training set TR and K target probabilities $\{\pi_j\}_{j=1}^K$ (line 2). The QuantTree algorithm returns a histogram defined by K bins $\{S_j\}_{j=1}^K$, where each $S_j \subset \mathbb{R}^d$ is set to contain $\pi_j N$ training samples. Further details on QuantTree – including how to define $\{(S_j, \pi_j)\}_{j=1}^K$ when TR cannot be exactly split to match target probabilities – can be found in [3].

The QT-EWMA statistic T_t monitors the proportion of samples in the datastream that fall in each bin S_j. In particular, for each x_t we define K binary statistics $\{y_{j,t}\}_j$ as the indicator functions of each bin S_j, namely

$$y_{j,t} = \mathbb{1}(x_t \in S_j), \quad j \in \{1, \dots, K\} \tag{3}$$

to track in which bin the input sample x_t falls. It is possible to show that, when $x_t \sim \phi_0$ and $TR \sim \phi_0$ then:

$$\mathbb{E}[y_{j,t}] \approx \hat{\pi}_j := \frac{N\pi_j}{N+1}, \quad j < K \quad \text{and} \quad \mathbb{E}[y_{K,t}] \approx \hat{\pi}_K := \frac{N\pi_K + 1}{N+1}. \tag{4}$$

We evaluate these statistics $y_{j,t}$ for each incoming sample x_t (line 5), and then we compute the EWMA statistic [28] $Z_{j,t}$, $j \in \{1, \dots, K\}$ (line 6), to monitor

Algorithm 1: QT-EWMA

```
   input  : datastream $x_1, x_2, \ldots$, target $\{\pi_j\}_{j=1}^K$, thresholds $\{h_t\}_t$, $TR$
   output : detection flag ChangeDetected, detection time $t^*$
1  ChangeDetected $\leftarrow$ False,   $t^* \leftarrow \infty$;
2  estimate QT histogram $\{(S_j, \pi_j)\}_{j=1}^K$ from $TR$ and define $\{\hat{\pi}_j\}_{j=1}^K$ as in (4);
3  $Z_{j,0} \leftarrow \hat{\pi}_j \; \forall j = 1, \ldots, K$;
4  for $t = 1, \ldots$ do
5      $y_{j,t} \leftarrow \mathbb{1}(x_t \in S_j)$;
6      $Z_{j,t} \leftarrow (1-\lambda) Z_{j,t-1} + \lambda y_{j,t}, \quad j = 1 \ldots, K$;
7      $T_t \leftarrow \sum_{j=1}^K (Z_{j,t} - \hat{\pi}_j)^2 / \hat{\pi}_j$;
8      if $T_t > h_t$ then
9          ChangeDetected $\leftarrow$ True,   $t^* \leftarrow t$;
10         break;
11     end
12 end
13 return ChangeDetected, $t^*$
```

the proportion of data that falls in each bin S_j:

$$Z_{j,t} = (1-\lambda) Z_{j,t-1} + \lambda y_{j,t} \quad \text{where} \quad Z_{j,0} = \hat{\pi}_j \, . \tag{5}$$

Since, under ϕ_0, the expected value $\mathbb{E}[Z_{j,t}] \approx \hat{\pi}_j$ for $j = 1, \ldots, K$, we define the QT-EWMA change-detection statistic as follows:

$$T_t = \sum_{j=1}^{K} \frac{(Z_{j,t} - \hat{\pi}_j)^2}{\hat{\pi}_j} \, . \tag{6}$$

Similarly to the Pearson statistic [20], T_t measures the overall difference between the proportion of points in each bin S_j, represented by $Z_{j,t}$, and their approximated expected values $\hat{\pi}_j$ under ϕ_0. This difference naturally increases as a consequence of a change $\phi_0 \to \phi_1$ that modifies the probability of some bin S_j. The QT-EWMA statistic is computed at each incoming sample (line 7) and then compared against the corresponding threshold h_t to detect changes (line 9).

The QT-EWMA algorithm inherits from QuantTree the fundamental property [3] that the distribution of the statistics (5) and (6) – like any other statistic entirely defined over QuantTree bins – does not depend on ϕ_0, so the thresholds $\{h_t\}_t$ can be defined *a priori* to guarantee the ARL_0 on any datastream.

4.2 Computing Thresholds Controlling the ARL_0

The sequence of thresholds $\{h_t\}_t$ has to be properly defined to guarantee a given ARL_0. According to the definition in Section 3, the $\text{ARL}_0 = \mathbb{E}[t^*]$, where the expected value is computed assuming that the whole datastream is drawn from ϕ_0. Following Margavio et al. [24], we adopt the relevant design choice of setting the thresholds $\{h_t\}_t$ to guarantee a constant false alarm probability α at each time instant t. When this property is satisfied, the detection time t^* is a Geometric random variable [24] with parameter α and expected value

$$\text{ARL}_0 = \mathbb{E}[t^*] = \frac{1}{\alpha}. \tag{7}$$

However, defining a set of thresholds $\{h_t\}_t$ that guarantee a constant false alarm probability α in online monitoring is not straightforward. As noted in [24], the thresholds must satisfy the following equation:

$$\mathbb{P}(T_t > h_t \mid T_k \leq h_k \ \forall k < t) = \alpha \quad \forall t \geq 1. \tag{8}$$

Since it is infeasible to exactly compute the conditional probabilities in (8), we resort to Monte Carlo simulations as in [29]. An important consequence of our design choice is that QuantTree properties [3] ensure that the thresholds for T_t do not depend on ϕ_0 nor on the data dimension d. Thus, we can conveniently generate 1-dimensional Gaussian streams and perform Monte Carlo simulations to compute the thresholds $\{h_t\}_t$ very efficiently.

In particular, we generate 1,000,000 training sets of $N = 4096$ normal realizations $x_t \sim \mathcal{N}(0,1)$. For each TR we construct a QuantTree histogram and then generate 5000 samples from $\mathcal{N}(0,1)$ that we use to compute the statistics $\{T_t\}_{t=1}^{5000}$ as in (6). Then, we define the threshold h_1 yielding the target ARL_0 as the empirical $(1-\alpha)$-quantile of T_1 values, bearing in mind that $\alpha = 1/\text{ARL}_0$. Similarly, all the thresholds h_t are computed as the $(1-\alpha)$-quantiles of the values T_t, but when $t > 1$ we compute the empirical quantiles only considering those sequences whose statistic has never exceeded any of the previous thresholds $h_1, \ldots, h_{t-1}$, namely having $T_k \leq h_k$,$\forall k < t$. Thresholds $\{h_t\}_t$ computed in this way guarantee (8) to hold, so the target ARL_0 is preserved.

We compute all the thresholds $\{h_t\}_{t=1}^{5000}$ and then fit a polynomial to these values, as suggested in [29]. This allows both to estimate h_t for $t > 5000$ and to improve the estimates $\{h_t\}_{t=1}^{5000}$ by leveraging correlation among thresholds. In particular, we estimate a polynomial in powers of $1/t$ that returns h_t for a given t. In our experiments we employ the following target ARL_0 values: 500, 1000, 2000, 5000, but we also compute and provide in the supplementary material the polynomial expressions of the thresholds for higher ARL_0 values (10000, 20000), which can be very useful to control false alarms in high-throughput applications.

An important consequence of setting a constant false alarm probability in (8) is that, being t^* a Geometric random variable with parameter α, the probability of having a false alarm before t by the geometric sum:

$$\mathbb{P}(t^* \leq t) = \sum_{k=1}^{t} \alpha(1-\alpha)^{k-1} = \alpha \cdot \frac{1-(1-\alpha)^t}{\alpha} = 1-(1-\alpha)^t, \tag{9}$$

where the probability $\mathbb{P}$ is computed under ϕ_0. This fact has relevant practical implications: since false alarms are controlled by (9), any substantial increase in their occurrence might indicate a drift in the data-generating process, which requires to update the change-detection algorithm. Other strategies based on asymptotic approximations cannot guarantee (9), as shown in our experiments.

5 Datastream Monitoring by One-shot Detectors

In this section we present how to adapt one-shot change-detection algorithms to monitor datstreams by controlling ARL_0. We focus on both algorithms that operate batch-wise (Section 5.1), and element-wise (Section 5.2).

5.1 Monitoring Datastreams by Batch-wise Detectors

Several change-detection algorithms operates dividing the datastream $x_1, x_2, \ldots$ in non-overlapping batches of ν samples:

$$W_t = [x_{(t-1)\nu+1}, \ldots, x_{t\nu}], \tag{10}$$

and computing a batch-wise test statistic $T^\nu(\cdot)$ over each W_t. This test statistic is typically defined upon a model fit over TR. For example, QuantTree [3] builds a histogram, while SPLL [17] fits a Gaussian Mixture $\widehat{\phi}_0$ to approximate ϕ_0. One-shot algorithms detect a change as soon as $T^\nu(W_t) > h^\nu$, and the threshold h^ν is set to control the probability of false alarm over each individual batch.

The very same monitoring scheme can be adopted to monitor datastreams at a controlled ARL_0, as long as the threshold h^ν is accordingly modified. This result is based on the following proposition

Proposition 1. *Let the threshold h^ν be such that*

$$\mathbb{P}(T^\nu(W) > h^\nu) = \alpha, \tag{11}$$

where W is a batch of ν samples drawn from ϕ_0. Then, the monitoring scheme $T^\nu(W) > h^\nu$ yields $ARL_0 \geq \nu/\alpha$.

Proof. Reported in the supplementary material due to space limitations.

Proposition 1 implies that, when we set $\alpha = \nu/\mathrm{ARL}_0$, our online change-detection algorithm is conservative, since its ARL_0 can be greater than the target. In some cases, however, we can provide guarantees that $\alpha = \nu/\mathrm{ARL}_0$, to exactly control the ARL_0, as shown in the following proposition.

Proposition 2. *Let the threshold h^ν be such that*

$$\mathbb{P}(T^\nu(W) > h^\nu \mid TR) = \alpha, \tag{12}$$

where W is a batch of ν samples drawn from ϕ_0. Then, the monitoring scheme $T^\nu(W) > h^\nu$ yields $ARL_0 = \nu/\alpha$.

Proof. Reported in the supplementary material due to space limitations.

We use the above propositions to adapt two relevant examples of batch-wise change-detection algorithm to monitor datastreams controlling ARL_0: QuantTree (QT) [3] and Semi-Parametric Log-Likelihood (SPLL) [17]. The theoretical property of QuantTree allows setting h^ν to guarantee (11) for any α regardless the data generating distribution ϕ_0. Thanks to Proposition 1, this can be extended to provide a lower bound on ARL_0 by instead setting the threshold h^ν as to guarantee that $\mathbb{P}(T^\nu(W) > h^\nu) = \nu/\text{ARL}_0$.

In the batch-wise SPLL detector, we compute the thresholds to guarantee the false positive rate using bootstrap over the training set TR since the distribution of the statistic depends on ϕ_0. In this case, we are under the hypothesis of Proposition 2, since the false positive probability is conditioned on the training set realization. Therefore, by adopting the same monitoring scheme and setting h^ν to guarantee $\mathbb{P}(T^\nu(W) > h^\nu | TR) = \nu/\text{ARL}_0$ we can obtain the target ARL_0. These two different guarantees will become very apparent in the experiments.

5.2 Monitoring Datastreams by Element-wise Detectors

As we remarked in Section 2, a popular approach to perform online monitoring in multivariate datastreams consists in reducing the dimensionality of the data, so that it is possible to monitor a 1-dimensional datastream using online change-detection algorithms designed for univariate data. Here we consider a dimensionality reduction method based on SPLL. In particular, we fit a Gaussian Mixture Model (GMM) $\widehat{\phi}_0$ on TR. Then, we reduce the dimensionality of each incoming sample x_t by computing $-\log(\widehat{\phi}_0(x_t))$, building a new, univariate sequence. Finally, we monitor this sequence using a nonparametric online CPM [29] leveraging the Lepage test statistic [21]. Remarkably, the resulting algorithm, which we call SPLL-CPM, operates at the target ARL_0 thanks to the CPM, whose thresholds have been computed for this purpose.

6 Computational Complexity

Here we analyze the computational complexity and memory requirements of QT-EWMA, Scan-B [22] (presented in Section 2), and of the modified one-shot detectors QuantTree [3], SPLL [17] and SPLL-CPM discussed in Section 5. The results of this analysis are summarized in Table 1.

QT-EWMA and QuantTree: Both algorithms are extremely efficient from both computational and memory points of view. Both place the incoming sample x_t in the corresponding bin of the QuantTree histogram, resulting in $\mathcal{O}(K)$ operations [3], where K is the number of bins. Then, QT-EWMA computes the test statistics (3), (5), (6) and these operations have a constant cost that falls within $\mathcal{O}(K)$. The QuantTree algorithm computes the Pearson statistic at the end of each batch, and this does not increase the order of computational complexity either, resulting in $\mathcal{O}(K)$ operations as in QT-EWMA. In terms of memory requirements, QT-EWMA updates the statistics $Z_{j,t}$ (5) at each new sample x_t, which requires storing only the K values $Z_{j,t-1}, j = 1, \ldots, K$. Similarly, the Pearson statistic in QuantTree requires storing the proportions of points from the batch falling in each of the K bins, thus K values. Hence, both algorithms have the same, constant, memory requirement of K values.

Table 1. Computational complexity for each update of the statistic and memory requirement of QT-EWMA compared to the other considered methods. In our experiments we set $K = 32$ for QT-EWMA and QT [3], $w = 1000$ for SPLL-CPM as in [29], and $B = 100, n = 5$ for Scan-B [22].

algorithm	QT-EWMA	QT [3]	SPLL [17]	SPLL-CPM	Scan-B[22]
complexity	$\mathcal{O}(K)$	$\mathcal{O}(K)$	$\mathcal{O}(md)$	$\mathcal{O}(md + w \log w)$	$\mathcal{O}(nBd)$
memory	K	K	1	w	$(n+1)Bd$

SPLL and SPLL-CPM Both these tests need to compute the log-likelihood of each sample with respect to each of the m components of a GMM $\widehat{\phi}_0$ fit on TR. This results in $\mathcal{O}(md)$ operations per sample [17]. In case of batch-wise monitoring (SPLL), averaging the log-likelihood over the batch, falls within the $\mathcal{O}(md)$ operations per sample, since it can be computed incrementally. Hence, only 1 value has to be stored. The SPLL-CPM algorithm instead leverages the CPM framework [29] to monitor the log-likelihood of each new observation x_t with respect to $\widehat{\phi}_0$. The Lepage statistic [21] used in the CPM requires sorting the entire sequence of log-likelihood values, which results in additional $\mathcal{O}(t \log t)$ operations. In this case, all the t values of the sequence have to be analyzed and stored, thus computational complexity and memory requirement steadily increase over time. Since this is not appropriate for online monitoring, [29] quantizes and stores old observations in a histogram, operating over a window of most recent w samples. Thus, both operations and memory requirements relate to w.
Scan-B The Scan-B algorithm [22] operates in a sliding-window fashion, with a fixed window size B and a fixed number n of reference windows from the training set. At each time step t, this algorithm requires to update n Gram matrices by computing $\mathcal{O}(B)$ times the MMD statistic, resulting in $\mathcal{O}(nBd)$ operations [16]. The Scan-B algorithm need to store n reference windows of B samples each from the training set, on top of the current window, yielding $(n + 1)Bd$ values in memory [16] for $d-$dimensional samples.

7 Experiments

Here we present the experimental evaluation of the proposed solution. Our aim is to show that QT-EWMA controls the false alarms better than competing methods while achieving lower or comparable detection delays. In QT-EWMA and QT we set $K = 32$ and uniform target probabilities $\pi_j = 1/K$, as in [3], since uniform histograms have been shown to be very effective for change detection purposes [4]. In both QT and SPLL we set $\nu = 32$, as in [3], and we employ the original configuration of the Scan-B algorithm [22].

7.1 Considered Datasets

We generate synthetic datastreams in different dimension $d \in \{2, 4, 8, 16, 32\}$ by choosing an initial Gaussian distribution ϕ_0 with random covariance matrix, and as alternative distribution a random roto-translation of ϕ_0 computed as $\phi_1 = \phi_0(Q \cdot + v)$. The roto-translation parameters Q and v are computed using the CCM framework [5] to guarantee a fixed symmetric Kullback-Leibler divergence $\mathrm{sKL}(\phi_0, \phi_1) \in \{0.5, 1, 1.5, 2, 2.5, 3\}$, which is useful to compare the detection performance obtained in different dimensions [1].

We also test our change-detection method on seven traditional multivariate classification datasets: Credit Card Fraud Detection ("credit", $d = 28$) from [6], Sensorless Drive Diagnosis ("sensorless", $d = 48$), MiniBooNE particle identification ("particle", $d = 50$), Physicochemical Properties of Protein Ternary Structure ("protein", $d = 9$), El Niño Southern Oscillation ("niño", $d = 5$), and two of the Forest Covertype datasets ("spruce" and "lodgepole", $d = 10$) from the UCI Machine Learning Repository [7]. As in [3], we standardize the datasets and sum to each component of "sensorless", "particle", "spruce" and "lodgepole" imperceptible Gaussian noise to avoid repeated values, which harm the construction of QuantTree histograms. The distribution of these datasets is typically considered stationary [18], so we randomly sample the datastreams from the datasets and introduce changes by shifting the post-change samples by a random vector drawn from a d-dimensional Gaussian scaled by the total variance of the dataset, as in [3, 18]. Here we report only the results on Gaussian datastreams with $d \in \{4, 32\}$ and the average results over these seven datasets (which we indicate by "UCI+credit"), leaving results on Gaussian data with $d \in \{2, 8, 16\}$ and over each individual dataset in the supplementary material.

We also test our method on the recently published INSECTS dataset [30] ($d = 33$), which describes the wing-beat frequency of different species of flying insects. This dataset is meant as a classification benchmark in datastreams affected by concept drift, i.e., the data is acquired under different environmental conditions affecting the insects' behavior. The dataset contains six concepts referring to different distributions. We assemble data from different concepts to form datastreams that include 30 realistic changes: we start sampling observations from one concept (ϕ_0) and switch to another (ϕ_1) introducing a change.

7.2 Figures of Merit

Empirical ARL_0. To assess whether QT-EWMA and the other considered methods maintain the target ARL_0, we compute the empirical ARL_0 as the average time before raising a false alarm. To this purpose, we run the considered methods on 5000 datastreams drawn from ϕ_0, setting the target $\mathrm{ARL}_0 \in \{500, 1000, 2000, 5000\}$. In this experiment, we consider datastreams of length $L = 6 \cdot \mathrm{ARL}_0$ to have a detection in each datastream. Since, by construction, the detection time t^* of our method under ϕ_0 is a Geometric random variable with parameter $\alpha = 1/\mathrm{ARL}_0$, (9) indicates that the probability of having a false alarm before L is $\mathbb{P}(t^* \leq L) \approx 0.9975$.
Detection delay. We evaluate the detection performance of QT-EWMA and the other considered methods by their detection delay, i.e. $\mathrm{ARL}_1 = \mathbb{E}[t^* - \tau]$, where the expectation is taken assuming that a change point τ is present [2]. We run the methods configured with target $\mathrm{ARL}_0 \in \{500, 1000, 2000, 5000\}$ on 1000

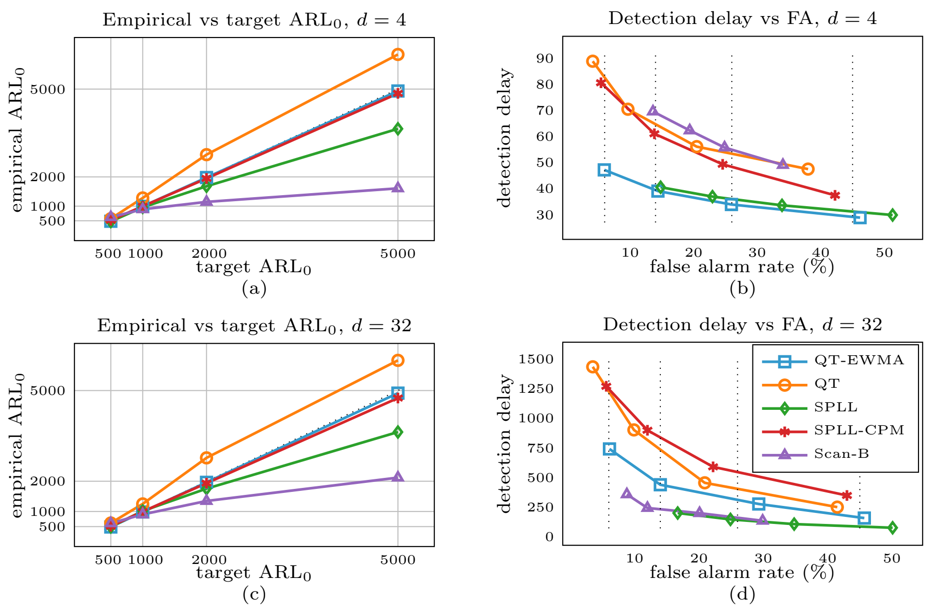

Fig. 1. Experimental results obtained on d-dimensional Gaussian datastreams ($d \in \{4, 32\}$). (a,c) show the empirical ARL_0 of the considered methods, and the target $ARL_0 \in \{500, 1000, 2000, 5000\}$ is maintained when the line is close to the dotted diagonal. (b,d) show the average detection delay against the percentage of false alarms, which should approach the dotted false alarm rates computed by (9).

datastreams of length 10000, each containing a change point at $\tau = 300$. We estimate the ARL_1 as the average difference $t^* - \tau$, excluding false alarms.
False alarm rate. To assess whether the considered methods maintain the target false alarm probability, we compute the percentage of false alarms obtained on the datastreams used to evaluate the detection delay, i.e., those in which a detection occurs at some $t^* < \tau$. Also in this case, we set the the target $ARL_0 \in \{500, 1000, 2000, 5000\}$, which according to (9) yield a false alarm in 45%, 26%, 14% and 6% of the datastreams, respectively.

7.3 Results and Discussion

Empirical ARL_0. The comparison of empirical and target ARL_0 on simulated Gaussian datastreams is reported in Figure 1(a,c), which show that QT-EWMA and SPLL-CPM control the ARL_0 very accurately, regardless of the data dimension, while the empirical ARL_0 of QT is higher than the target, as we expected from Proposition 1. Figures 2(a,c) show that we obtain the same result on datastreams sampled from the considered real-world dataset, confirming the non-parametric nature of QuantTree. In contrast, SPLL and Scan-B cannot control high target values of ARL_0 (on both simulated and real-world datastreams), due

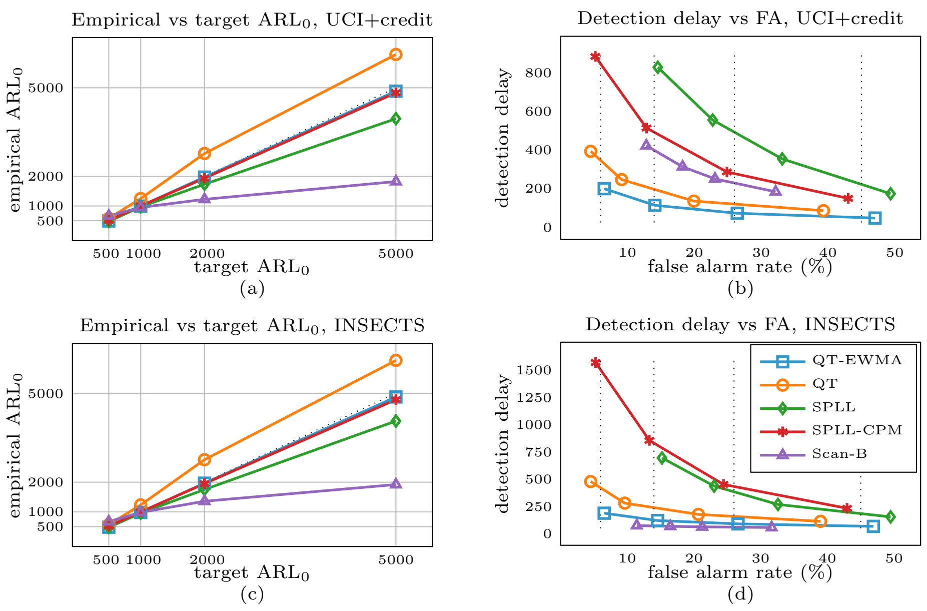

Fig. 2. Experimental results obtained on datastreams sampled from real-world datasets. (a,c) show the empirical ARL$_0$ of the considered methods respectively on the UCI+credit and INSECTS datasets. The target ARL$_0 \in \{500, 1000, 2000, 5000\}$ is maintained when the line is close to the dotted diagonal. (b,d) show the detection delay against the percentage of false alarms achieved on the same datasets, which should approach the dotted false alarm rates by (9).

to detection thresholds inaccurately estimated. In particular, the SPLL statistic strongly depends on ϕ_0, so the thresholds have to be computed by bootstrap on a relatively small training set, which might yield inaccurate estimates. Thresholds for Scan-B are defined by an asymptotic approximation that strongly depends on the sliding window size B [22]. In particular, this approximation is more accurate for higher target ARL$_0$ when B is large, which increases the computational complexity and memory usage (Table 1). Thus, Scan-B with a fixed window size B can only maintain low ARL$_0$ values.

Detection delay vs false alarms. We plot the percentage of false alarms against the average detection delay, setting different ARL$_0$ values, to assess the trade-off between these two quantities. Figures 1(b,d) show the performance of the considered methods on simulated Gaussian datastreams ($d \in \{4, 32\}$, respectively) containing a change point at $\tau = 300$ such that sKL$(\phi_0, \phi_1) = 2$.

In terms of detection delay, QT-EWMA is the best method when $d = 4$, while SPLL outperforms all the others when $d = 32$, which is expected since its parametric assumptions are met (ϕ_0 is a Gaussian). All methods decrease their power as d increases, which is also expected due to *detectability loss* [1]. Scan- B seems to be more robust to detectability loss, yielding lower detection

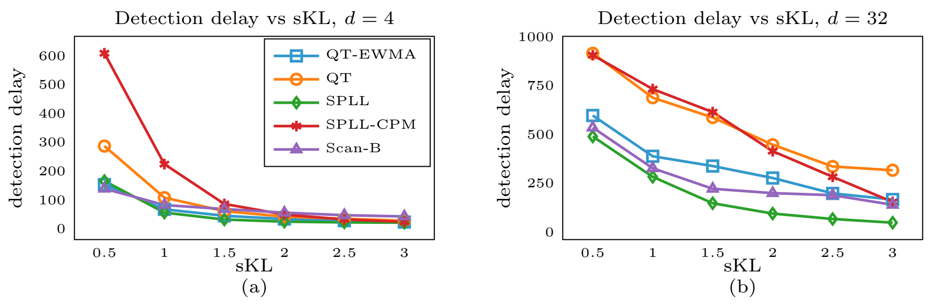

Fig. 3. Detection delay as a function of the change magnitude computed on simulated Gaussian datastreams with dimension $d \in \{4, 32\}$ containing a change point at $\tau = 300$ with target $\text{ARL}_0 = 1000$ (which is maintained by all methods, see Figure 1(a,c)).

delays than QT-EWMA on Gaussian data when d increases, as shown in Figures 1(b,d) and in supplementary material. Statistics defined on histograms are known to be less powerful than those based on MMD, as they perceive only changes affecting bin probabilities (and are for instance totally blind to distribution changes inside each bin). This effect can be mitigated by increasing the number of bins, thus the computational complexity, which is reasonable when d increases (e.g., when $K = d = 32$ there is on average a single split per dimension). However, in the considered real-world datasets, QT-EWMA turns out to be more effective than Scan-B in even larger dimensions, as shown in Figures 2(b,d) and supplementary material. On average, QT-EWMA is clearly the best method in terms of detection delay on the UCI+credit datasets, and achieves delays similar to Scan-B on the INSECTS dataset. The fact that QT-EWMA consistently outperforms QT on simulated and real-world data indicates that our sequential statistic is more powerful than the original QuantTree statistic (designed for batch-wise monitoring) when detecting changes online.

In terms of false alarm rate, QT-EWMA and SPLL-CPM approach the target values computed by (9), while QT and SPLL have fewer and more false alarms, respectively, as a consequence of their empirical ARL_0, and this happens in all the considered monitoring scenarios. The false alarms of Scan-B, instead, exhibit a completely different behavior which also depends on the data distribution, due to the fact that its thresholds do not yield a constant false alarm probability.

Detection delay. Figure 3 shows the detection delays of the considered methods as a function of the change magnitude on Gaussian datastreams. To enable a fair comparison, we configured all methods by setting the target $\text{ARL}_0 = 1000$, which all methods can maintain (see Figures 1(a,c)). The best method on Gaussian datastreams is SPLL, as can be expected since its parametric assumptions are met. The best nonparametric method is QT-EWMA when $d = 4$, and Scan-B when $d = 32$. As observed also in Figure 1(b,d), Scan-B seems to be more robust to detectability loss than QT-EWMA. As expected, all methods decrease their detection delays when the change magnitude increases.

8 Conclusions

We introduce QT-EWMA, a novel nonparametric online change-detection algorithm for multivariate datastreams. Our monitoring scheme is computationally light and can effectively maintain the target ARL_0 and the expected false alarm rates. Such accurate control over false alarms is very useful in practical applications. Our experiments on simulated and real-world data show that alternative solutions do not provide such guarantees in nonparametric settings, and that QT-EWMA turns out to be very effective in real-world datasets. Future work concerns incremental monitoring schemes based on QuantTree, to start monitoring with small training sets, and further investigation on how to set the optimal number of bins in QuantTree histograms for online change detection.

References

1. Alippi, C., Boracchi, G., Carrera, D., Roveri, M.: Change detection in multivariate datastreams: Likelihood and detectability loss. Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) **2**, 1368–1374 (2016)
2. Basseville, M., Nikiforov, I.V., et al.: Detection of abrupt changes: theory and application, vol. 104. Prentice Hall Englewood Cliffs (1993)
3. Boracchi, G., Carrera, D., Cervellera, C., Macciò, D.: QuantTree: histograms for change detection in multivariate data streams. In: International Conference on Machine Learning. pp. 639–648 (2018)
4. Boracchi, G., Cervellera, C., Macciò, D.: Uniform histograms for change detection in multivariate data. In: 2017 IEEE International Joint Conference on Neural Networks (IJCNN). pp. 1732–1739. IEEE (2017)
5. Carrera, D., Boracchi, G.: Generating high-dimensional datastreams for change detection. Big data research **11**, 11–21 (2018)
6. Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., Bontempi, G.: Credit card fraud detection: a realistic modeling and a novel learning strategy. IEEE Transactions on Neural Networks and Learning Systems **29**(8), 3784–3797 (2017)
7. Dua, D., Graff, C.: UCI machine learning repository (2017), `http://archive.ics.uci.edu/ml`
8. Fellouris, G., Tartakovsky, A.G.: Multichannel sequential detection—part I: Non-i.i.d. data. IEEE Transactions on Information Theory **63**(7), 4551–4571 (2017)
9. Frittoli, L., Bocchi, M., Mella, S., Carrera, D., Rossi, B., Fragneto, P., Susella, R., Boracchi, G.: Strengthening sequential side-channel attacks through change detection. IACR Transactions on Cryptographic Hardware and Embedded Systems **3**, 1–21 (2020)
10. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM computing surveys (CSUR) **46**(4), 44 (2014)
11. Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., Smola, A.J.: A kernel method for the two-sample-problem. In: Advances in Neural Information Processing Systems. pp. 513–520 (2007)
12. Guralnik, V., Srivastava, J.: Event detection from time series data. In: 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 33–42 (1999)
13. Hawkins, D.M., Qiu, P., Kang, C.W.: The changepoint model for statistical process control. Journal of quality technology **35**(4), 355–366 (2003)

14. Ho, S.S.: A martingale framework for concept change detection in time-varying data streams. In: Proceedings of the 22nd International Conference on Machine Learning. pp. 321–327 (2005)
15. Hotelling, H.: A generalized t test and measure of multivariate dispersion. In: Proceedings of the second Berkeley symposium on mathematical statistics and probability. University of California (1951)
16. Keriven, N., Garreau, D., Poli, I.: NEWMA: a new method for scalable model-free online change-point detection. IEEE Transactions on Signal Processing (2020)
17. Kuncheva, L.I.: Change detection in streaming multivariate data using likelihood detectors. IEEE Transactions on Knowledge and Data Engineering **25**(5), 1175–1180 (2011)
18. Kuncheva, L.I., Faithfull, W.J.: PCA feature extraction for change detection in multidimensional unlabeled data. IEEE Transactions on Neural Networks and Learning Systems **25**(1), 69–80 (2013)
19. Lau, T.S., Tay, W.P., Veeravalli, V.V.: A binning approach to quickest change detection with unknown post-change distribution. IEEE Transactions on Signal Processing **67**(3), 609–621 (2018)
20. Lehmann, E.L., Romano, J.P.: Testing statistical hypotheses. Springer (2006)
21. Lepage, Y.: A combination of Wilcoxon's and Ansari-Bradley's statistics. Biometrika **58**(1), 213–217 (1971)
22. Li, S., Xie, Y., Dai, H., Song, L.: M-statistic for kernel change-point detection. Advances in Neural Information Processing Systems **28**, 3366–3374 (2015)
23. Lung-Yut-Fong, A., Lévy-Leduc, C., Cappé, O.: Robust changepoint detection based on multivariate rank statistics. In: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 3608–3611. IEEE (2011)
24. Margavio, T.M., Conerly, M.D., Woodall, W.H., Drake, L.G.: Alarm rates for quality control charts. Statistics & Probability Letters **24**(3), 219–224 (1995)
25. Mei, Y.: Efficient scalable schemes for monitoring a large number of data streams. Biometrika **97**(2), 419–433 (2010)
26. Mozafari, N., Hashemi, S., Hamzeh, A.: A precise statistical approach for concept change detection in unlabeled data streams. Computers & Mathematics with Applications **62**(4), 1655–1669 (2011)
27. Qahtan, A.A., Alharbi, B., Wang, S., Zhang, X.: A PCA-based change detection framework for multidimensional data streams. In: 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 935–944 (2015)
28. Roberts, S.: Control chart tests based on geometric moving averages. Technometrics **1**(3), 239–250 (1959)
29. Ross, G.J., Tasoulis, D.K., Adams, N.M.: Nonparametric monitoring of data streams for changes in location and scale. Technometrics **53**(4), 379–389 (2011)
30. Souza, V.M.A., Reis, D.M., Maletzke, A.G., Batista, G.E.A.P.A.: Challenges in benchmarking stream learning algorithms with real-world data. Data Mining and Knowledge Discovery pp. 1–54 (2020)
31. Tartakovsky, A.G., Rozovskii, B.L., Blazek, R.B., Kim, H.: A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods. IEEE Transactions on Signal Processing **54**(9), 3372–3382 (2006)
32. Xie, Y., Siegmund, D.: Sequential multi-sensor change-point detection. In: 2013 Information Theory and Applications Workshop. pp. 1–20. IEEE (2013)
33. Zamba, K., Hawkins, D.M.: A multivariate change-point model for statistical process control. Technometrics **48**(4), 539–549 (2006)