

# Continuous monitoring for changepoints in data streams using adaptive estimation

Dean A. Bodenham · Niall M. Adams

Received: date / Accepted: date

**Abstract** Data streams are characterised by a potentially unending sequence of high frequency observations which are subject to unknown temporal variation. Many modern streaming applications demand the capability to sequentially detect changes as soon as possible after they occur, while continuing to monitor the stream as it evolves. We refer to this problem as *continuous monitoring*. Sequential algorithms such as CUSUM, EWMA and their more sophisticated variants usually require a pair of parameters to be selected for practical application. However, the choice of parameter values is often based on the anticipated size of the changes and a given choice is unlikely to be optimal for the multiple change sizes which are likely to occur in a streaming data context. To address this critical issue we introduce a changepoint detection framework based on adaptive forgetting factors that, instead of multiple control parameters, only requires a single parameter to be selected. Simulated results demonstrate that this framework has utility in a continuous monitoring setting. In particular, it reduces the burden of selecting parameters in advance. Moreover, the methodology is demonstrated on real data arising from Foreign Exchange markets.

## 1 Introduction

Modern data acquisition technology has provided the opportunity to reason about streaming data (e.g. Gama 2010; Aggarwal 2006). Such data are often characterised by a potentially unending sequence of observations, subject to *a priori* unknown temporal variation. Typically, streaming data is observed at a high frequency with respect to the computational demands of the analysis tools deployed against it. This combination of characteristics means that streaming analysis has two specific demands: the need for efficient computation and the need for a mechanism to handle unknown temporal variation, as it happens. Streaming analysis of computer network traffic (e.g. Bodenham and Adams 2013, 2014) is a topical example. Typical of streaming data applications, the flow of traffic data continues *uninterrupted* as it is monitored even if changes are flagged.

Adaptive estimation (e.g. Haykin 2002) is a simple approach for handling temporal variation, where a parameter  $\lambda$ , called a *forgetting factor*, is used to smoothly down-weight historical data as new data arrives. In certain cases, in particular for the exponential family of distributions, this approach yields a computationally-efficient implementation. Specifically, such implementations need only examine each streaming datum once, and have a constant and low memory demand. Moreover, other ideas from adaptive filter theory can yield *automatic* sequential selection of the forgetting factor (Anagnostopoulos et al 2012). The capability to automatically set such control parameters is crucial for streaming applications, since human intervention is impractical.

Adaptive estimation approaches are not primarily intended for explicit change detection, but instead for keeping an estimator close to a time-varying target.

---

Dean A. Bodenham  
Department of Mathematics, Imperial College London  
D-BSSE, ETH Zürich  
Tel.: +44 (020) 7594 8837  
Fax: +44 (020) 7594 8517  
E-mail: d.bodenham10@imperial.ac.uk

Niall M. Adams  
Department of Mathematics, Imperial College London  
Heilbronn Institute of Mathematics, University of Bristol

This methodology has been successfully deployed, or adapted, to a number of streaming data problems (Anagnostopoulos et al 2012; Pavlidis et al 2011; Adams et al 2010). In this paper, adaptive estimation is used to provide an up-to-date estimator that will provide some resilience to the errors that can occur in change detection, such as false positives and missed detections.

This research is concerned with the particular scenario of sequentially detecting multiple changepoints in discrete-time, univariate streaming data, which we refer to as *continuous monitoring*. This scenario is rather different from the more conventional problem of statistical process control, where a detected change could result in, for example, a manufacturing device being stopped for corrective intervention. Instead, the data stream continues to flow, uninterrupted by any detected changes. To re-emphasise an earlier point, the character of the changes in a data stream is *unknown*. An exemplar application, explored in Ross et al (2011, Sec. 4), arises in monitoring financial time series. Here, the value of a financial instrument is subject to change as a result of diverse forces in the market, but the value continues to evolve. A single trader detecting changes in the stream might result in a trading action, which may indirectly affect the stream, but does not fundamentally stop it. Similar problems occur in certain types of security and surveillance applications (Frisén 2003).

*Continuous monitoring* thus refers to the problem of sequentially detecting changes in an unending sequence of data, where the character of the changes is unknown (location, size and type of change, as discussed further below) and, moreover, the change detector must be able to restart following a detected change, and repeat this process indefinitely. This challenging problem is the subject of the present paper. There are numerous issues to consider, including the definitions of a successfully detected change, the mechanics of restarting, and the crucial problem of selecting control parameters for the change detector.

We restrict attention in this article to continuous monitoring of the *mean* of a data stream, where the *location* and *size* of changes are unknown. While this framework can, in principle, be used to reason about any parameter, we restrict attention to monitoring the mean to focus on the challenges of the streaming scenario. In this context, beyond computational issues, there are three methodological problems to address. First is the need to handle changes of unknown and unpredictable size. Second is the need to handle restarting, wherein the detector needs to begin again after a change is detected. Third is the need to set the detector's control parameters *repeatedly and without expert interaction* as the stream evolves. As discussed in Section 3.3,

the problem of unknown change sizes has been explored extensively, particularly via so-called adaptive-CUSUM and adaptive-EWMA procedures. Similarly, the problem of restarting has also received study. Notably, there is little evidence of intersection between these two threads of literature, i.e. methods that handle both restarting and unknown change sizes. Methods developed for either of the former problems still suffer from the requirement of parameters that are difficult to set without some knowledge of the expected behaviour of the stream.

Our interest in deploying adaptive estimation methodology is partially motivated by the need to specify all control parameters in advance, which is clearly impossible for the continuous monitoring problem. The remaining motivation relates to how such a detector will respond to changes. A detector using adaptive estimation should be more capable of recovering from a false positive than a conventional restarting sequential change detector. Usually, a change is detected after some delay, or may be missed entirely; however, a forgetting factor deals with this gracefully by having the estimate automatically adapt to the new regime whether or not a change is detected. We restrict attention to the i.i.d. context, either because it is natural in the context of the data, or alternatively because we are monitoring the residuals from a model which would capture any dependence in the observations.

This introduction has sought to raise a number of issues arising in the continuous monitoring problem that simply do *not* arise in conventional sequential change detection. The remainder of this paper is structured as follows: Section 2 introduces the continuous monitoring problem. Section 2.2 discusses performance measures relevant to continuous monitoring. A review of sequential change detection methods is provided in Section 3 with special focus on the most basic formulations of CUSUM and EWMA. Section 3.3 discusses more sophisticated variants on these basic methods. Having set up this background, Section 4 describes adaptive estimation using a forgetting factor, either fixed or adaptive. A simulation study, comparing adaptive estimation with restarting CUSUM and EWMA in a continuous monitoring context, is presented in Section 5. The simulation results require careful analysis, but suggest that adaptive estimation has merit for continuous monitoring. In Section 6, we demonstrate our adaptive estimation change detection methodology in an application related to financial data.

## 2 Continuous monitoring

Usually change detection algorithms are compared by their ability to find a single changepoint, with the pre-change and post-change distributions possibly unknown. However, in many real-world situations such as financial monitoring (exemplified in Section 6), multiple changepoints are expected and an algorithm must continue to monitor the process for successive changes. In this section we discuss the multiple changepoint scenario and relevant performance metrics.

Denote a stream of observations as  $x_1, x_2, \dots$ , sampled from i.i.d. random variables  $X_1, X_2, \dots$ , with changepoints  $\tau_1, \tau_2, \dots$ , such that

$$\begin{aligned} X_1, X_2, \dots, X_{\tau_1} &\sim F_1, \\ X_{\tau_1+1}, X_{\tau_1+2}, \dots, X_{\tau_2} &\sim F_2, \\ X_{\tau_2+1}, X_{\tau_2+2}, \dots, X_{\tau_3} &\sim F_3, \text{ etc,} \end{aligned} \quad (1)$$

where  $F_1, F_2, \dots$  represent distributions such that  $F_k \neq F_{k+1}$  for all  $k$ . For a change in the mean, the size of the  $i$ th change is  $|E[X_{\tau_i+1}] - E[X_{\tau_i}]|$  for  $i \geq 1$ .

### 2.1 Restarting: the need to estimate parameters

For most parametric sequential change detection algorithms even if, in order to detect changepoint  $\tau_i$ , the parameters for the *post-change* distribution  $F_{i+1}$  are not required in order to detect a change, parameters for the *pre-change* distribution  $F_i$  will almost certainly be needed. These parameters are often unknown in practical problems. One solution is to assume that, at the start of the  $i$ th regime after changepoint  $\tau_{i-1}$ , the process is *in-control* for a certain number of observations,  $x_{\tau_i}, x_{\tau_i+1}, \dots, x_{\tau_i+B}$ , and then use these observations to estimate the parameters for the current regime (Jones et al 2001; Jones 2002; Jones et al 2004). This parameter estimation stage is called the *burn-in* period. For all the algorithms deployed below we estimate both the mean and variance of the stream during the burn-in period. This version of change detection, where the unknown stream parameters are estimated, is particularly challenging. However, issues arise with this restarting approach, since if the burn-in is not sufficiently long to estimate the parameters accurately, the performance of any change detection algorithm will be severely affected (Jensen et al 2006).

Note that while there are sequential change detection algorithms that do not require estimates of the pre-change parameters (Appel and Brandt 1983; Hawkins et al 2003), these algorithms are all concerned with detecting a single change, and not multiple changes, which is the main focus of this paper.

### 2.2 Performance measures

For the single changepoint scenario, the Average Run Lengths ARL0 and ARL1 (described in Section 2.2.1) are sufficient to describe the performance of a change detection algorithm. However, assessment of performance becomes complicated once we depart from the most basic sequential change detection setting (i.e. a single change in univariate stream). For example, in the context of a multivariate change detection problem, Sullivan (2002) developed extra performance measure.

Performance assessment is complicated in the continuous monitoring problem, and extends beyond the standard approaches used in the literature. We consider first the conventional metrics, and then performance metrics relevant to the continuous monitoring scenario.

#### 2.2.1 Average Run Length

Two standard performance measures are the Average Run Lengths, ARL0 and ARL1 (Page 1954). ARL0 is computed as the average number of observations until a changepoint is detected, when the algorithm is run over a sequence of observations with no changepoints, while ARL1 is the average number of observations between a changepoint occurring and the change being detected. Note that ARL1 typically refers to a single change of a given magnitude.

As noted in the introduction, the challenge of continuous monitoring involves a sequence of changes of unknown and varying magnitude. These measures alone are insufficient to characterise detection performance in a continuous monitoring framework. Issues related to calculating the ARLs in a continuous monitoring setting are discussed in Section 5.2.

#### 2.2.2 Detection rates

The ARL1 value neither reflects how many changepoints are detected nor how many are missed. Moreover, ARL1 and ARL0 together do not reflect the ratio of true detections to false positives. In a single-change context, these might be difficult to measure, since any reasonable algorithm will detect a change given enough time. However, in a data stream, there is a finite amount of time between changepoints, and some changes might not be detected before another changepoint occurs, and we then classify these as *missed changes*.

Now, suppose that we have a data stream with  $C$  changepoints, and our algorithm makes a total of  $D$  detections,  $T$  of which are true (correct) detections, while  $D - T$  are false detections. We then define:

- $CCD = T/C$ , the proportion of *changepoints* correctly detected
- $DNF = T/D$ , the proportion of *detections* that are not false detections

These intuitive definitions are the same as *sensitivity* and *positive predicted value (PPV)* in the surveillance literature (German et al 2001; Fraker et al 2008), and are equivalent to *recall* and *precision*, respectively, in the pattern recognition literature. Similar metrics are discussed in Kifer et al (2004).

Although the “complements” of  $CCD$  and  $DNF$  may be more intuitively defined (proportions of missed changepoints and false detections, respectively), these definitions are preferred since the closer  $CCD$  and  $DNF$  are to 1, the better the performance of the algorithm.

### 3 Review of Sequential Change Detection

This section provides an overview of the standard CUSUM and EWMA procedures, in addition to more sophisticated variants. The overview relates to the context of a stream of observations  $x_1, x_2, \dots$ , sampled independently from the random variables  $X_1, X_2, \dots$  which have mean  $\mu$  and variance  $\sigma^2$ . If these parameters are unknown, as occurs with streaming data, estimates are used, as discussed in Section 2.1.

#### 3.1 CUSUM

The Cumulative Sum (CUSUM) algorithm was first proposed in Page (1954) and requires knowledge of the pre- and post-change distributions of the stream for optimality (Moustakides 1986), in the sense of Lorden (1971). However, if the distributions’ parameters are unknown, as is often the case, then this optimality is not guaranteed.

If the stream is initially  $N(\mu, \sigma^2)$ -distributed, the CUSUM statistics  $S_j$  and  $T_j$  are defined as:  $S_0 = T_0 = \mu$ , and

$$\begin{aligned} S_j &= \max(0, S_{j-1} + (x_j - \mu)/\sigma - k), & j \in \{1, 2, \dots\} \\ T_j &= \max(0, T_{j-1} - (x_j - \mu)/\sigma - k), & j \in \{1, 2, \dots\} \end{aligned}$$

in order to detect an increase or decrease in the mean. A change is detected when either  $S_j > h$  or  $T_j > h$ .

Here the *control parameters*  $k$  and  $h$  need to be chosen. These values are often chosen according to the needs of the application, and specifically the magnitude of the changes one is trying to detect; a selection of recommended values for  $k$  and  $h$  are provided in Hawkins (1993). This selection is based on the anticipated change size,  $|E[X_{\tau_i+1}] - E[X_{\tau_i}]|$ . In continuous

monitoring where changes of unknown size will occur, setting these parameters in this manner is unrealistic because the change point size cannot be anticipated. Handling the problem of setting control parameters repeatedly in the continuous monitoring context is a key concern of this paper.

#### 3.2 EWMA

The Exponentially Weighted Moving Average (EWMA) scheme was first described in Roberts (1959) and is defined by the statistic  $Z_j$ :  $Z_0 = \mu$ , and

$$Z_j = (1 - r)Z_{j-1} + rx_j, \quad j \in \{1, 2, \dots\},$$

where  $r$  is a chosen control parameter. The standard deviation of  $Z_j$  is

$$\sigma_{Z_j} = \sigma \sqrt{\frac{r}{2-r} [1 - (1-r)^{2j}]},$$

and a change is detected when either  $Z_j > \mu + L\sigma_{Z_j}$  or  $Z_j < \mu - L\sigma_{Z_j}$ , where  $L$  is a second control parameter that needs to be chosen. Again, setting values for the control parameters  $r$  and  $L$  is somewhat subjective, or based on desired ARL0 performance, with Lucas and Saccucci (1990) suggesting values in the range  $r \in [0.05, 1.0]$ , and  $L \in [2.4, 3.0]$ .

#### 3.3 More sophisticated approaches

CUSUM and EWMA have been detailed above because they are the two most basic and well-studied approaches for change detection. Of course, many sophisticated variations have been proposed, each of which typically handle only one of the the challenges in continuous monitoring. Re-capping the requirements of continuous monitoring provides a convenient way to partition the relevant literature:

- R1** Sequential and efficient computation
- R2** Handling changes of unknown size
- R3** Few control parameters

**R2** has been studied extensively in the context of a *single* changepoint. Much of this work is related to so-called adaptive-CUSUM and adaptive-EWMA; see Tsung and Wang (2010) for a review. Note that we are not aware of any literature where both **R2** and restarting are addressed together.

Apley and Chin (2007) provides an optimal filtering mechanism which reduces to standard EWMA in special cases. The approach is shown to be effective for both large and small changes. However, this approach

is inadequate for continuous monitoring due to **R1** and **R3**, specifically the large number of coefficients to be estimated in the filter.

In addition to addressing **R2**, Capizzi and Masarotto (2012) proposes a method that is suitable for different size shifts when there is dependence in the process generating the post-change distribution, in the context of a single change. This sophistication comes at some computational cost, which make this approach unsuitable for continuous monitoring in relation to **R1** and **R3**.

Again, with respect to **R2**, Jiang et al (2008) propose a hybrid EWMA/CUSUM procedure in the context of a single change. While this approach looks effective in experiments, there are four parameters to be determined (**R3**), which would be challenging in the continuous monitoring context where parameters cannot be routinely specified after any change. Similarly, Capizzi and Masarotto (2003) suggests an adaptive EWMA scheme which requires three control parameters.

This issue of self-starting has been addressed in both univariate and multivariate contexts. For example, an early approach to multivariate self-starting can be found in Hawkins (1987). This method has two parameters, the setting of which is suggested by reference to standard tables, such as those in Lucas (1976). Other approaches to self-starting include Sullivan (2002) and Xie and Sigmund (2013). In all these examples, one way or another, there are multiple control parameter settings (**R3**) that are challenging in the context of continuous monitoring.

Finally, while there are approaches for detecting multiple changepoints in a stream (e.g. Xie and Sigmund (2013); Maboudou-Tchao and Hawkins (2013)), in general these are either non-sequential (Maboudou-Tchao and Hawkins 2013, **R1**) or require several control parameters (Xie and Sigmund 2013, **R3**).

This review of previous work has sought to show that while there are sophisticated methods designed to deal with individual aspects of the continuous monitoring scenario, these methods do not meet the minimum requirements for deployment in a continuous monitoring context. For this reason, the experimental study of Section 5 involves a comparison between adaptive estimation-based change detection and basic restarting CUSUM and EWMA. This is deliberately the simplest comparison possible, in order to explore the continuous monitoring problem in detail.

#### 4 Adaptive estimation using a forgetting factor

We start by describing an adaptive estimation scheme to monitor the mean of a stream of observations with

a fixed forgetting factor  $\lambda$ . Next we extend this to an adaptive forgetting factor  $\vec{\lambda}$ , and then we define a decision rule for deciding when a change has occurred.

##### 4.1 Estimating the mean with a fixed forgetting factor

Suppose we have a data stream  $x_1, x_2, \dots$ , and we have seen  $N$  observations so far. The arithmetic mean of these observations is defined as

$$\bar{x}_N = \frac{1}{N} \sum_{i=1}^N x_i. \quad (2)$$

In this sum each observation is given equal weight, namely  $1/N$ . The rationale behind adaptive estimation is that, for a time-varying process, we would like to place more weight on more recent observations, since this better reflects the current regime of the data stream. This logic is central to adaptive estimation, as typified by Haykin (2002). In adaptive estimation, we introduce an exponential *forgetting factor*  $\lambda \in [0, 1]$  and define the *forgetting factor mean*  $\bar{x}_{N,\lambda}$  as

$$\bar{x}_{N,\lambda} = \frac{1}{w_{N,\lambda}} \sum_{i=1}^N \lambda^{N-i} x_i, \quad w_{N,\lambda} = \sum_{i=1}^N \lambda^{N-i}.$$

Observe that if  $0 < \lambda < 1$ , then earlier observations are down-weighted by higher powers of  $\lambda$ , and so more weight is placed on later observations. Alternatively, we can define  $\bar{x}_{N,\lambda} = \frac{m_{N,\lambda}}{w_{N,\lambda}}$  for  $N \geq 1$ , and

$$m_{N,\lambda} = \lambda m_{N-1,\lambda} + x_N, \quad w_{N,\lambda} = \lambda w_{N-1,\lambda} + 1 \quad (3)$$

with  $m_{0,\lambda} = w_{0,\lambda} = 0$ . Note that  $\lambda = 0$  and  $\lambda = 1$  are degenerate cases. Setting  $\lambda = 0$  corresponds to forgetting all previous observations, and only using the most recent observation, i.e.  $\bar{x}_{N,0} = x_N$ . On the other hand,  $\lambda = 1$  corresponds to no forgetting, and then the forgetting factor mean is simply the usual arithmetic mean, i.e.  $\bar{x}_{N,1} = \bar{x}_N$ .

The reader may observe that these equations bear some resemblance to the EWMA equations above. Indeed, they are related; using the above equations we can rewrite

$$\begin{aligned} \bar{x}_{N,\lambda} &= \left(1 - \frac{1}{w_{N,\lambda}}\right) \bar{x}_{N-1,\lambda} + \frac{1}{w_{N,\lambda}} x_N \\ &= \lambda \left(\frac{1 - \lambda^{N-1}}{1 - \lambda^N}\right) \bar{x}_{N-1,\lambda} + \frac{1 - \lambda}{1 - \lambda^N} x_N, \end{aligned} \quad (4)$$

and then if  $\lambda \in (0, 1)$ , as  $N \rightarrow \infty$ , this becomes

$$\bar{x}_{N,\lambda} = \lambda \bar{x}_{N-1,\lambda} + (1 - \lambda) x_N,$$

which is equivalent to the EWMA scheme if we set  $r = 1 - \lambda$  (Choi et al 2006). However, there will be differences for finite  $N$ . Additionally, the forgetting factor formulation allows for the easily defined adaptive estimation scheme discussed in Section 4.2, which partially reduces the burden of selecting parameters.

The relationship between this adaptive forgetting factor framework and the Kalman filter provides a partial theoretical justification for the approach. Suppose a random walk is characterised for  $i = 1, 2, \dots$  by

$$\begin{aligned} X_i &\sim N(\mu_i, \sigma_X^2), \\ \mu_i &= \mu_{i-1} + \xi_i, \\ \xi_i &\sim N(0, \sigma_\xi^2). \end{aligned}$$

for some parameters  $\sigma_X$  and  $\sigma_\xi$ . In Anagnostopoulos (2010, Sec. 3.1.6) it is remarked that for such a random walk, if the parameters  $\sigma_X$  and  $\sigma_\xi$  are known then the optimal filter estimate after observation  $N$ , given all observations  $x_1, x_2, \dots, x_N$ , is denoted by

$$\hat{\mu}_N^{KF} = E[X_N | x_1, \dots, x_N; \sigma_X, \sigma_\xi],$$

and then  $\hat{\mu}_N^{KF}$  is recursively computable by the Kalman filter equations (Kalman 1960). Furthermore, it is shown in Anagnostopoulos (2010, Sec. 3.1.6) that (for this special case)  $w_0^{KF} = \hat{\mu}_0^{KF} = 0$ , and for  $N > 0$

$$\hat{\mu}_N^{KF} = \left(1 - \frac{1}{w_N^{KF}}\right) \hat{\mu}_{N-1}^{KF} + \left(\frac{1}{w_N^{KF}}\right), \quad (5)$$

$$w_{N+1}^{KF} = \lambda_N w_N^{KF} + 1, \quad (6)$$

$$\lambda_N = \left[ \frac{\sigma_\xi^2}{\sigma_X^2} w_N^{KF} + 1 \right]^{-1} \quad (7)$$

It is interesting to compare Equations (3) and (4) for the fixed forgetting factor mean estimator with Equations (5) and (6). This comparison shows that the optimal filter equations are of the same form as the forgetting factors equations. However, it is apparent from this comparison that the optimal forgetting factor for the Kalman filter is intimately related to the signal-to-noise ratio,  $\sigma_\xi^2/\sigma_X^2$ .

## 4.2 Defining an adaptive forgetting factor

It is not clear how to select a value  $\lambda \in [0, 1]$  for the fixed forgetting factor scheme above. Indeed, it is likely that no single parameter setting will be optimal, as we claim is the case with CUSUM and EWMA in continuous monitoring. It is also conceivable that the performance of the estimator could be improved if  $\lambda$  were allowed to vary as the stream develops. For instance, after a sudden change the forgetting factor should be closer to

zero, to quickly “forget” the previous regime, and when in-control it should be closer to 1. Adaptive (variable) forgetting factors were first explored in Åström and Wittenmark (1973) and Åström et al (1977), and later in Fortescue et al (1981). We define the *adaptive forgetting factor* (AFF)  $\vec{\lambda}$ , as the sequence  $\vec{\lambda} = (\lambda_1, \lambda_2, \dots)$ . In what follows we shall introduce an optimisation parameter, which we address later, and the value of which will be shown to be unimportant. Motivated by Equation (3), we define the *adaptive forgetting factor mean*  $\bar{x}_{N, \vec{\lambda}}$  as follows:

**Definition 1** For a sequence of observations  $x_1, x_2, \dots, x_N$ , and forgetting factors  $\lambda_1, \lambda_2, \dots, \lambda_N$ , after defining

$$\begin{aligned} m_{N+1, \vec{\lambda}} &= \lambda_N m_{N, \vec{\lambda}} + x_{N+1}, & m_{0, \vec{\lambda}} &= 0, \\ w_{N+1, \vec{\lambda}} &= \lambda_N w_{N, \vec{\lambda}} + 1, & w_{0, \vec{\lambda}} &= 0, \end{aligned} \quad (8)$$

the *adaptive forgetting factor mean*  $\bar{x}_{N, \vec{\lambda}}$  is defined, for  $N \geq 1$ , by  $\bar{x}_{N, \vec{\lambda}} = \frac{m_{N, \vec{\lambda}}}{w_{N, \vec{\lambda}}}$ .

Henceforth, note that  $\vec{\lambda}$  in a subscript indicates that an adaptive forgetting factor is used, while  $\lambda$  indicates a fixed forgetting factor. When  $\vec{\lambda}$  is used for  $m_{N, \vec{\lambda}}$ , all  $\lambda_i \in (\lambda_1, \lambda_2, \dots, \lambda_{N-1})$  are implicitly used in its calculation.

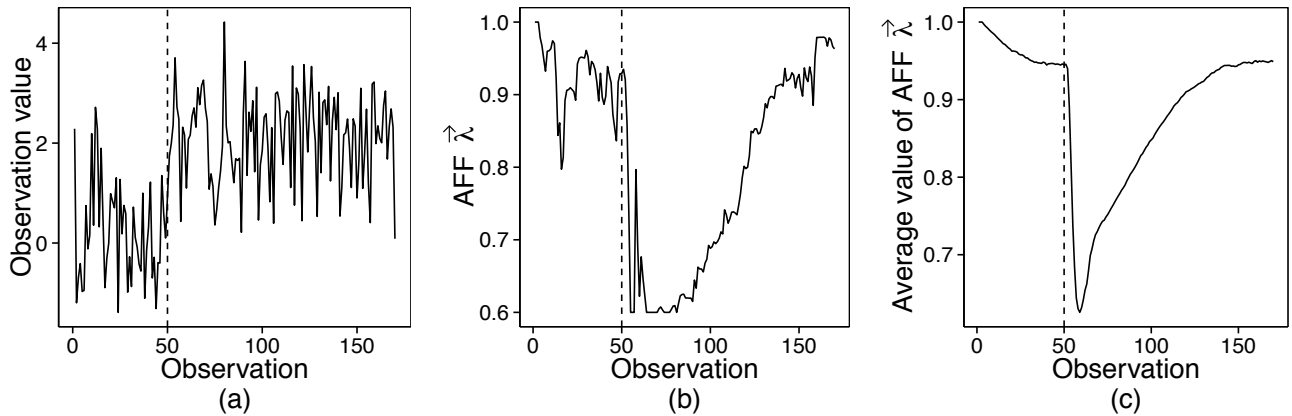
Determining how to update  $\lambda_N \rightarrow \lambda_{N+1}$  is the key question when using an adaptive forgetting factor. Following ideas in Haykin (2002, Chap. 14), we update  $\vec{\lambda}$  using a two-step process. We first choose a cost function  $L_{N, \vec{\lambda}}$  of  $\bar{x}_{N, \vec{\lambda}}$ , and then update  $\vec{\lambda}$  in the direction that minimizes  $L_{N, \vec{\lambda}}$  by

$$\lambda_N = \lambda_{N-1} - \eta \frac{\partial}{\partial \vec{\lambda}} L_{N, \vec{\lambda}}, \quad (9)$$

where  $\eta \ll 1$ . This is a variant of stochastic gradient descent (e.g. Borkar 2008). Note that the derivative with respect to  $\vec{\lambda}$  is a special scalar-valued derivative which will be described in detail in Section 4.2.1 below. We defer a discussion on the value of  $\eta$  to Section 4.4, where it will be shown that for a broad range of values the change detection method seems insensitive to the selection of  $\eta$ . Note that the choice of the cost function  $L_{N, \vec{\lambda}}$  is context-dependent, and there are many possible choices for this function. In the next section, our choice of  $L_{N, \vec{\lambda}}$  will be motivated by our desire to monitor the *mean* of a sequence of observations.

### 4.2.1 The derivative with respect to $\vec{\lambda}$

The derivative of a function with respect to  $\vec{\lambda}$  can be defined as follows:



**Fig. 1** Stream  $x_1, x_2, \dots$  sampled from  $X_k \sim N(0,1)$  for  $k \leq 50$ , and  $X_k \sim N(1,1)$  for  $k > 50$ . Value of observations (left), value of AFF  $\vec{\lambda}$  over one stream (middle) and value of AFF  $\vec{\lambda}$  averaged over 1000 such streams (right). In each case, the step size  $\eta = 0.01$ .

**Definition 2** For any function  $f_{N, \vec{\lambda}}$  involving  $\vec{\lambda}$ ,

$$\frac{\partial}{\partial \vec{\lambda}} f_{N, \vec{\lambda}} = \lim_{\epsilon \rightarrow 0} \frac{1}{\epsilon} [f_{N, \vec{\lambda} + \epsilon} - f_{N, \vec{\lambda}}],$$

where  $\vec{\lambda} + \epsilon = (\lambda_1 + \epsilon, \lambda_2 + \epsilon, \dots)$ .

As a particular example, which will be needed later, consider the derivative of  $m_{N, \vec{\lambda}}$ . Although it is defined sequentially in Equation (8), it can be written in a non-sequential form<sup>1</sup>, according to the following proposition, proved in the Supplementary Material (Sec. 1.1):

**Proposition 3** For  $N = 1, 2, \dots$ ,  $m_{N, \vec{\lambda}}$  defined sequentially in Definition 1 can be computed using:

$$m_{N, \vec{\lambda}} = \sum_{i=1}^N \left[ \left( \prod_{p=i}^{N-1} \lambda_p \right) x_i \right]. \quad (10)$$

Using this proposition, we then have:

$$m_{N, \vec{\lambda} + \epsilon} = \sum_{i=1}^N \left[ \left( \prod_{p=i}^{N-1} (\lambda_p + \epsilon) \right) x_i \right] \quad (11)$$

$$= (\lambda_{N-1} + \epsilon) m_{N-1, \vec{\lambda} + \epsilon} + x_N, \quad (12)$$

and so we define its derivative as  $\Delta_{N, \vec{\lambda}}$ :

$$\Delta_{N, \vec{\lambda}} = \frac{\partial}{\partial \vec{\lambda}} m_{N, \vec{\lambda}} = \lim_{\epsilon \rightarrow 0} \frac{1}{\epsilon} [m_{N, \vec{\lambda} + \epsilon} - m_{N, \vec{\lambda}}]. \quad (13)$$

To get an expression for  $\Delta_{N, \vec{\lambda}}$ , we need the following result, proved in the Supplementary Material (Sec. 1.2):

<sup>1</sup> Note that the empty product has value:  $\prod_{p=N}^{N-1} \lambda_p = 1$ .

**Lemma 4** For  $\lambda_i, \lambda_{i+1}, \dots, \lambda_M$ ,  $i \geq 1$ , and  $\epsilon \ll 1$ ,

$$\prod_{t=i}^M (\lambda_t + \epsilon) = \prod_{t=i}^M \lambda_t + \epsilon \left( \sum_{t=i}^M \left( \prod_{\substack{p=i \\ p \neq t}}^M \lambda_p \right) \right) + O(\epsilon^2). \quad (14)$$

This result, combined with Equations (10)-(13), allows us to find a non-sequential form for  $\Delta_{N, \vec{\lambda}}$ :

**Proposition 5** Following the definition of  $\Delta_{N, \vec{\lambda}}$  in Equation (13),  $\Delta_{N, \vec{\lambda}}$  can be computed using:

$$\Delta_{N, \vec{\lambda}} = \sum_{i=1}^{N-1} \left[ \sum_{t=i}^{N-1} \left( \prod_{\substack{p=i \\ p \neq t}}^{N-1} \lambda_p \right) x_i \right]. \quad (15)$$

This result, proved in the Supplementary Material (Sec. 1.3-1.4), leads to the sequential update equation

$$\Delta_{N+1, \vec{\lambda}} = \lambda_N \Delta_{N, \vec{\lambda}} + m_{N, \vec{\lambda}}, \quad \Delta_{1, \vec{\lambda}} = 0. \quad (16)$$

Similarly we obtain for  $\Omega_{N, \vec{\lambda}} = \frac{\partial}{\partial \vec{\lambda}} w_{N, \vec{\lambda}}$ :

$$\Omega_{N+1, \vec{\lambda}} = \lambda_N \Omega_{N, \vec{\lambda}} + w_{N, \vec{\lambda}}, \quad \Omega_{1, \vec{\lambda}} = 0. \quad (17)$$

Interestingly, Equations (16) and (17) agree with those given in Anagnostopoulos et al (2012). However, there the derivative was defined by assuming  $\lambda_{i+1} = \lambda_i$ , which is a counter-intuitive assumption to make in order to derive update equations for *time-varying*  $\vec{\lambda}$ . Here we have derived the update equations, Equations (16) and (17), without needing to make such an assumption. Indeed, the derivation above rigorously confirms the heuristic argument in Anagnostopoulos et al (2012).

#### 4.2.2 The choice of cost function $L_{N, \vec{\lambda}}$

Equations (16) and (17) allow us to sequentially compute the derivative of any well-behaved cost function  $L_{N, \vec{\lambda}}$  which is a function of  $w_{N, \vec{\lambda}}$  and  $m_{N, \vec{\lambda}}$ . For example, defining

$$L_{N, \vec{\lambda}} = [\bar{x}_{N-1, \vec{\lambda}} - x_N]^2, \quad (18)$$

which turns out to be an appropriate choice when adaptively estimating the mean of a sequence of observations, the derivative is simply (see Supplementary Material, Sec. 1.5)

$$\frac{\partial}{\partial \vec{\lambda}} L_{N, \vec{\lambda}} = 2[\bar{x}_{N-1, \vec{\lambda}} - x_N] \left[ \frac{\Delta_{N-1, \vec{\lambda}} - \bar{x}_{N-1, \vec{\lambda}} \Omega_{N-1, \vec{\lambda}}}{w_{N-1, \vec{\lambda}}} \right] \begin{cases} E[\bar{X}_{N, \vec{\lambda}}] = \mu, & \text{Var}[\bar{X}_{N, \vec{\lambda}}] = (u_{N, \vec{\lambda}}) \sigma^2, \\ \text{where } u_{1, \vec{\lambda}} = 1 \text{ and, for } i \geq 1, \end{cases}$$

The choice of cost function  $L_{N, \vec{\lambda}}$  in Equation (18) can be motivated by a sequential maximum likelihood formulation for i.i.d. normal observations (Anagnostopoulos et al 2012), and for the rest of the article this cost function will be used.

Figure 1 demonstrates the behaviour of an adaptive forgetting factor. The left frame gives some simulated data, where the process exhibits a simple jump in the mean. The middle frame displays the corresponding value of the adaptive forgetting factor. The right frame shows the average behaviour of the adaptive forgetting factor, over 1000 random realizations of the process. We see that the forgetting factor exhibits desirable behaviour, specifically that  $\vec{\lambda}$  exhibits a marked drop immediately after the change, and then later recovers to its previous level after a period of stability. Note that after updating  $\vec{\lambda}$  according to Equation (9), it is possible that  $\lambda_i \notin [0, 1]$ . In order to ensure  $\lambda_i \in [0, 1]$ , after updating  $\vec{\lambda}$  we use, for some  $\lambda_{min} \in [0, 1]$ :

$$\lambda_i = \max \{ \min \{ \lambda_i, 1 \}, \lambda_{min} \},$$

In fact, we use  $\lambda_{min} = 0.6$ , rather than  $\lambda_{min} = 0$ , so that  $\vec{\lambda}$  does not get too close to 0 and recovers faster after a change to pre-change levels. Section 1.6 in the Supplementary Material provides justification for this choice.

We finally note that while there have been attempts to create an adaptive EWMA chart (e.g. Capizzi and Masarotto 2003), these are of a fundamentally different character and use score functions which smoothly interpolate a EWMA and Shewhart chart. Now that the AFF estimation framework has been fully described, we discuss a decision rule for detecting a change.

#### 4.3 Deciding when a change has occurred

The adaptive estimation scheme described above provides a computationally efficient means to provide an up-to-date estimate of the mean of the stream. Extra reasoning is required to extend to a change detection framework, as now discussed. First, we need the following proposition, proved in the Supplementary Material (Sec. 2.1):

**Proposition 6** *If our data stream is sampled from the i.i.d. random variables  $X_1, X_2, \dots, X_N$ , with  $E[X_i] = \mu$  and  $\text{Var}[X_i] = \sigma^2$  for all  $i \geq 1$ , then  $\bar{X}_{N, \vec{\lambda}}$ , the adaptive forgetting factor mean of  $X_1, X_2, \dots, X_N$ , defined according to Definition 1, has expectation and variance*

$$u_{i+1, \lambda} = \left( 1 - \frac{1}{w_{i+1, \lambda}} \right)^2 u_{i, \lambda} + \left( \frac{1}{w_{i+1, \lambda}} \right)^2.$$

Note that Proposition 6 makes no mention of normality, or the distribution of the  $X_k$  variables, simply their expectation and variance.

While monitoring our forgetting factor mean  $\bar{x}_{N, \lambda}$ , we need a decision rule to decide that a change has occurred. A number of possibilities are available for reasoning about the distribution of  $\bar{X}_{N, \lambda}$ . Since both the mean and variance are available, a simple approach based on Chebyshev's inequality is possible. However, experiments with this approach indicate that the inequality is not sufficiently tight to yield good change detection performance, and so it leads to challenges in automatically setting the control parameter. Following much of the development of the literature in sequential analysis (Gustafsson 2000; Basseville and Nikiforov 1993), we assume that the distributions generating our observations are normal.

If we assume that the pre-change distribution is normal, specifically  $D_1 = N(\mu, \sigma^2)$ , then we have from the above that  $\bar{X}_{N, \lambda} \sim N(\mu, (u_{N, \lambda}) \sigma^2)$ . Then, for a given  $\alpha \in [0, 1]$ , we can use the normal cdf to find limits  $L_1, L_2$  such that

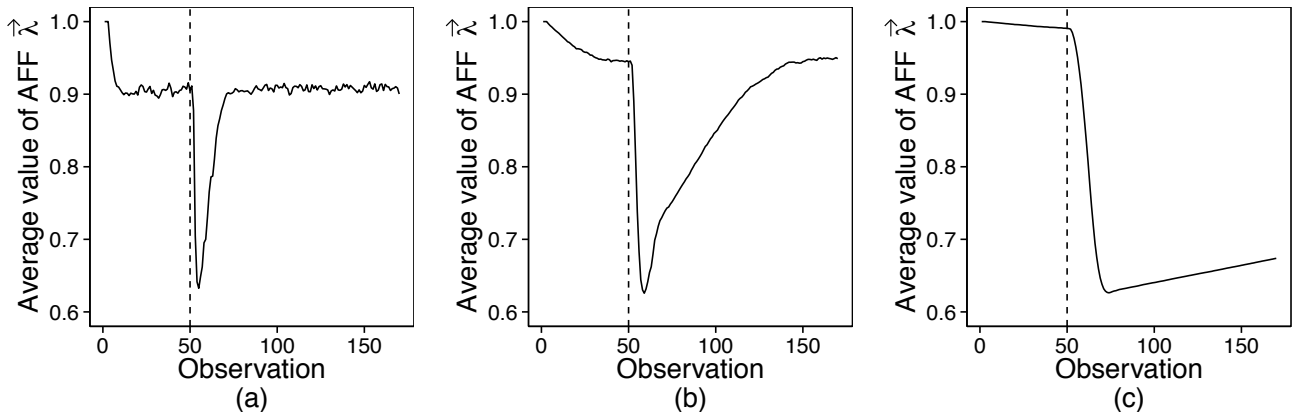
$$\Pr [\bar{X}_{N, \lambda} \leq L_1] = \alpha/2, \quad \Pr [\bar{X}_{N, \lambda} \geq L_2] = 1 - \alpha/2,$$

and our *control limits* are then defined by the interval  $P_{N, \lambda, \alpha} = (L_1, L_2)$ . A change is detected if  $\bar{x}_{N, \lambda} \notin P_{N, \lambda, \alpha}$ . The choice of  $\alpha$  is, of course, application-specific. So, for our AFF scheme, we only need to choose one control parameter, the significance level  $\alpha$ . This is different to CUSUM and EWMA which each require two control parameters to be chosen.



Algo	Parameter	Value	Step size $\eta$	CCD	DNF	ARL1 (SDRL1)	ARL0 (SDRL0)
AFF	$\alpha$	0.005	0.1	0.85	0.82	27.23 (32.27)	670.04 (1018.23)
AFF	$\alpha$	0.005	0.01	0.86	0.79	27.12 (32.05)	819.36 (1162.97)
AFF	$\alpha$	0.005	0.001	0.86	0.78	24.89 (29.65)	987.68 (1336.78)

**Table 1** This table shows the detection performance for AFF with parameter  $\alpha = 0.05$  and step size  $\eta \in \{0.1, 0.01, 0.001\}$ , for a stream with approximately 5000 changepoints of size  $\delta$ , where  $\delta$  is sampled uniformly from  $\{0.25, 0.5, 1, 3\}$ , and using burn-in length  $B=50$ . This table shows that the performance of AFF is very similar for a range of values for  $\eta$ . The performance measures CCD, the proportion of chnangepoints correctly and detected, and DNF, the proportion of detected changepoints that are not false detections, are defined in Section 2.2.2. The Average Run Lengths, ARL0 and ARL1, are defined in Section 2.2.1, while the SDRL0 and SDRL1 are their standard deviations, respectively.



**Fig. 2** The average behaviour of the AFF  $\hat{\lambda}$  for a single change, as in Figure 1 (right-most figure). Left, middle and right figures here use  $\eta = 0.1, 0.01, 0.001$ , respectively. Each scheme reacts to the change quickly, but recovery is slow when  $\eta = 0.001$ .

#### 4.4 Choice of step size $\eta$

Although the AFF algorithm change detector only relies on a single parameter  $\alpha$  being chosen, we should wonder if the step size (learning rate)  $\eta$  in Equation (9) affects its performance. Indeed, there is very little guidance in the literature on setting this gradient descent parameter. In this section we explore the impact of choosing different values for  $\eta$ .

Before looking at values at  $\eta$ , we first take a closer look at the scale of the derivative  $\frac{\partial}{\partial \lambda} L_{k, \vec{\lambda}}$  in Equation (9). First, we arrive at the following result, proved in the Supplementary Material (Sec. 2.2):

**Proposition 7** *If our data stream is sampled from the i.i.d. random variables  $X_1, X_2, \dots, X_N$ , with  $E[X_i] = \mu$  and  $\text{Var}[X_i] = \sigma^2$  for all  $i \geq 1$ , then*

$$E \left[ \frac{\partial}{\partial \vec{\lambda}} L_{N, \vec{\lambda}} \right] \sim O(\sigma^2),$$

where  $L_{N, \vec{\lambda}}$  is defined as in Equation (18).

This result is important, since  $\vec{\lambda}$  is a value in the range  $[0, 1]$ , and if  $\sigma^2$  is too large, the gradient descent step in Equation (9) will force  $\vec{\lambda}$  to be either 0 or 1, depending on whether the derivative is positive or negative, respectively. A simple approach to remedying this issue is to scale the derivative by  $\sigma^2$ . We therefore scale

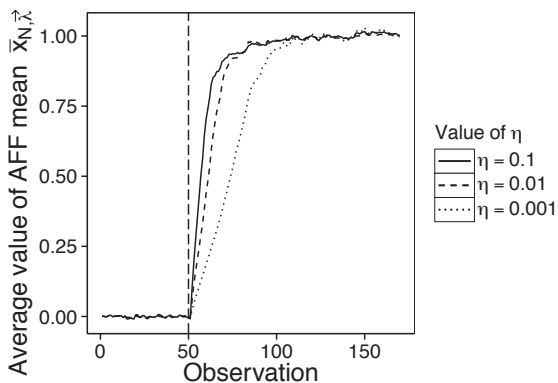
$\frac{\partial}{\partial \vec{\lambda}} L_{N, \vec{\lambda}}$  by the estimate of the variance obtained during the burn-in period,  $\hat{\sigma}^2$ , i.e.

$$\lambda_N = \lambda_{N-1} - \eta \left( \frac{1}{\hat{\sigma}^2} \right) \frac{\partial}{\partial \vec{\lambda}} L_{N, \vec{\lambda}}, \quad (19)$$

In other words, this data-driven procedure removes the burden of scaling the derivative. With this taken care of, we investigate the performance of the AFF change detector for  $\eta = 0.1, 0.01, 0.001$ . Table 1 below shows that, when considering a data stream containing thousands of changepoints with a variety of change sizes, the AFF algorithm performs relatively consistently for  $\eta = 0.1, 0.01, 0.001$ , using the approach above of scaling the derivative. Further tables, showing similarly stable performance for when the change sizes are the same, are included in the Supplementary Material (see Sec. 3, specifically Sec. 3.3).

Figure 2 shows the value of  $\vec{\lambda}$  drops after a change, regardless of the value of  $\eta$ . However, the smaller the value of  $\eta$ , the slower  $\vec{\lambda}$  recovers to its pre-change levels. Schemes could be explored where  $\vec{\lambda}$  is reset to 1 after a change has been detected, but such refinements are not investigated here.

Figure 3 shows that the value  $\eta$  does not substantially affect the average value of  $\bar{x}_{N, \vec{\lambda}}$ . While there is a difference immediately following the changepoint at



**Fig. 3** The impact of the step size  $\eta$  on the AFF mean  $\bar{x}_{N,\lambda}$ , for data as in Figure 1. Each line represents the average of the AFF mean  $\bar{x}_{N,\lambda}$  over 1000 simulations.

$\tau = 50$ , by observation 100 the average values seem to be identical. In particular, the behaviour of  $\bar{x}_{N,\lambda}$  for  $\eta = 0.1$  and  $\eta = 0.01$  is very similar.

## 5 Simulation study

In developing new change detection methodology, it is customary to consider the case of normally-distributed data (e.g. Capizzi and Masarotto 2010; Hawkins 1987). This simulation study follows this custom, specifying  $F_i \sim N(\mu_i, \sigma_i)$ , and additionally  $\sigma_i = 1$ , for all  $i$ .

Randomly spaced changepoints are now generated, following the scheme illustrated in Figure 4(a), by first sampling  $\xi_i, \xi_2, \dots \sim \text{Pois}(\nu)$ , for some value  $\nu$ , to obtain random interval widths. These  $\xi_i$  are then padded with values  $G$  and  $D$ ;  $G$  is a *grace* period to give the algorithm time to estimate the streams parameters, and  $D$  is a period that allows the algorithm to detect a change. The changepoints are therefore specified by:

$$\begin{aligned} \tau_1 &= G + \xi_1 \\ \tau_i &= \tau_{i-1} + D + G + \xi_i, \quad i \in \{2, 3, \dots, M\}. \end{aligned}$$

The stream is then generated in blocks  $[\tau_i+1, \tau_{i+1}]$ . The first block is sampled from a normal distribution with mean  $\mu_1 = 0$ , and then block  $i$  is sampled with mean  $\mu_i = \mu_{i-1} + \theta \cdot \delta_i$ , where  $\delta_i$  is a random jump size in some set  $S$ , and  $\theta$  is uniformly sampled from  $\{+1, -1\}$  to allow for increases and decreases in the mean. For the simulations below, the stream is generated with parameters

$$\nu = 50, \quad G = 50, \quad D = 50, \quad M = 5000,$$

and the set of jump sizes  $\delta_i$  is uniformly sampled from  $S = \{0.25, 0.5, 1, 3\}$ .

This simulation investigates the performance of CUSUM, EWMA and AFF. Furthermore, following a suggestion from an anonymous reviewer, we also investigate the performance of the method described in Jiang et al (2008), which we abbreviate to JSA (based on the authors' names). An overview of the JSA method can be found in the Section 6 in the Supplementary Material. All algorithms use a burn-in period of length  $B$  to estimate the parameters of the stream (mean and variance) before monitoring for the changepoint.

Note that our stream is generated to contain 5000 changepoints in order to obtain accurate values for ARL1, DNF and CCD. In the standard single changepoint setting, this would be done by running 1000 trials, where each trial measures the ability to detect a single changepoint. In our case, since we have different changepoint sizes, we estimate these quantities by evaluating over a single stream with many changepoints. Note that ARL0 is computed in the standard way; 1000 trials are run, where each trial consists of the algorithm attempting to find a changepoint on a long stationary stream (all observations generated from the same distribution). Any changepoint that is detected is a false detection, and the delay to detecting that false changepoint contributes to the ARL0.

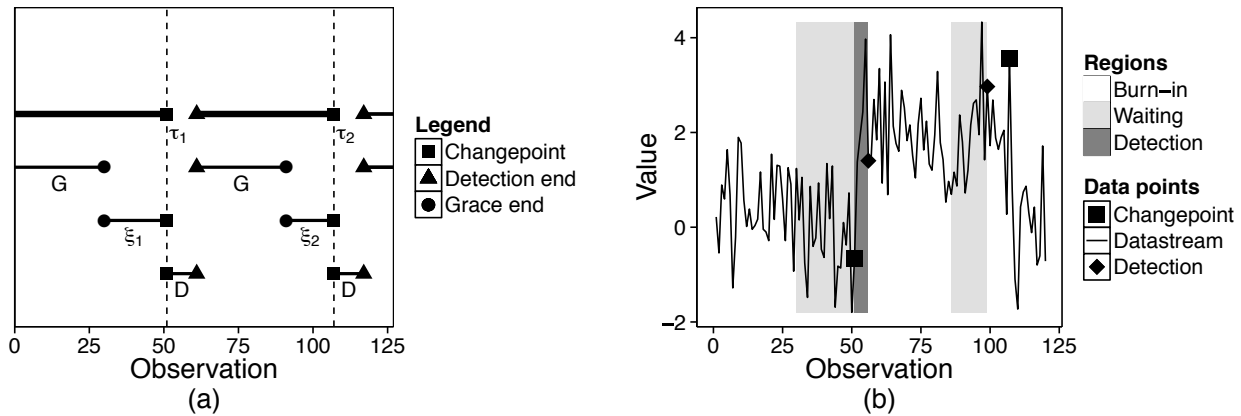
### 5.1 Classifying the detected changes

After running over the stream, an algorithm will return a sequence of detected changepoints  $\{\hat{\tau}_1, \hat{\tau}_2, \dots\}$ , and we must then classify these as correct or false detections. In a simulation setting, we would use the sequence of true changepoints  $\{\tau_1, \tau_2, \dots\}$  to do this, and would also record which of these true changepoints were missed.

Suppose that we wish to classify the detected changepoint  $\hat{\tau}_n$ . After the previous changepoint  $\hat{\tau}_{n-1}$ , a change detection algorithm in our simulation would have used the period  $[\hat{\tau}_{n-1} + 1, \hat{\tau}_{n-1} + B]$  to estimate the mean and variance of the stream (Note: for  $n = 1$ ,  $\tau_0 = 0$ ). Suppose the next *true* changepoint that occurs after  $\hat{\tau}_{n-1} + B$  is  $\tau_m$ , and that the following true changepoint is  $\tau_{m+1}$ . We now have three possible scenarios:

- if  $\hat{\tau}_n \in [\hat{\tau}_{n-1} + B, \tau_m]$ , then  $\hat{\tau}_n$  is a *false detection*,
- if  $\hat{\tau}_n \in [\tau_m + 1, \tau_{m+1}]$ , then  $\hat{\tau}_n$  is a *correct detection*,
- if  $\hat{\tau}_n > \tau_{m+1}$ , then  $\tau_m$  is a *missed detection*.

In the case of a missed detection,  $\hat{\tau}_n$  is taken to be the correct detection of the true changepoint directly preceding it, e.g.  $\tau_{m+1}$ . In order to visualise the situation better, one can imagine that our stream is divided into three regions of different coloured backgrounds, *burn-in*, *waiting*, and *detection* regions. Then, a detected



**Fig. 4** (a) Generating the stream, as described in Section 5. The values  $\tau_1, \tau_2, \dots$  are the changepoint locations, based on the random interval widths  $\xi_1, \xi_2, \dots$  padded by a grace period  $G$  and a period  $D$  to allow for the detection of the changepoint. Changepoint locations further highlighted by vertical dashed lines. (b) Schematic representation of detection regions, as described in Section 5.1. A changepoint is correctly detected if a detection is made in a *detection region*, while a detection in a *waiting region* indicates a false detection.

changepoint  $\hat{\tau}_n$  is classified according to the region in which it lies. For example, in Figure 4(b) the first detected changepoint is a correct detection, while the second detected changepoint is a false detection.

## 5.2 Average run length for a data stream

The calculations of ARL0 and ARL1 are simple using this framework. The ARL1 is the sum of the lengths of the detection regions between the correctly detected changepoints and their nearest true changepoints. Note that this excludes the detection regions that are between two true changepoints (missed detections).

While it is tempting to calculate ARL0 as the sum of the lengths of the waiting regions between false detections (divided by the number of false detections), we instead follow the traditional approach for calculating ARL0. That is, for  $i = 1, 2, \dots, M$  trials, the algorithm is run over a sufficiently long stationary stream (i.e. without changepoints) until a change is falsely detected at  $\hat{\tau}_i$ . The ARL0 is then computed to be  $\frac{1}{M} \sum_{i=1}^M \hat{\tau}_i$ .

The variances of ARL0 and ARL1 are also calculated, and their standard deviations SDRL0 and SDRL1, respectively, are reported. These values are known as the *standard deviations of the run lengths*.

## 5.3 Results

Table 2 displays exemplar results for CUSUM, EWMA and AFF algorithms for a stream with over 750,000 observations containing over 5000 changepoints with a variety of change sizes, in an effort to simulate a real-world data stream.

The parameter values used for CUSUM and EWMA are the recommended choices in Hawkins (1993) and Lucas and Saccucci (1990), respectively. However, the performance of CUSUM and EWMA can vary dramatically depending on the choice of parameter values. Part of the reason for this is that these different parameter choices are the suggested choices when trying to detect a *single* change of a known magnitude. This is clearly not the case in a streaming data context where multiple changepoints of *different* magnitudes should be expected. Therefore, it is a serious drawback to need to select one of these parameter choices, when one cannot know what change-size to expect. Even if all the changes were of the same magnitude, as long as this magnitude were unknown one would still not be able to make an informed choice of parameter values. The Supplementary Material (Sec. 3.1) contains tables showing similarly diverse performance (for CUSUM and EWMA) even when the changes in the stream are all of the same magnitude. Finally it is worth mentioning that, to the best of our knowledge, no previous work has investigated the performance of CUSUM and EWMA operating on a stream with multiple, different-sized changes.

In the case of AFF, however, performance is far more stable. It was shown in Section 4.4 and Table 2 that the value of the step size  $\eta$  is not important, since a variety of  $\eta$  values lead to very similar performance. Further tables in the Supplementary Material (Sec. 3.3) reinforce this point. Therefore, one is left to select the value of the single control parameter  $\alpha$ . Indeed, setting  $\alpha$  is a simple matter of deciding on the width of the control limits, e.g. as in the construction of a confidence interval,  $\alpha = 0.01$  for a 99% interval. Clearly, larger values of  $\alpha$  lead to narrower intervals and more sensitive

Algo	Parameters	Values	CCD	DNF	ARL1	(SDRL1)	ARL0	(SDRL0)
CUSUM	$(k, h)$	(1.50, 1.61)	0.65	0.80	30.55	(36.86)	534.54	(890.49)
CUSUM	$(k, h)$	(1.25, 1.99)	0.70	0.80	29.91	(36.57)	538.56	(903.96)
CUSUM	$(k, h)$	(1.00, 2.52)	0.75	0.79	27.14	(34.13)	524.86	(905.54)
CUSUM	$(k, h)$	(0.75, 3.34)	0.80	0.79	26.08	(32.28)	454.60	(795.29)
CUSUM	$(k, h)$	(0.50, 4.77)	0.85	0.77	24.73	(30.52)	373.38	(612.43)
CUSUM	$(k, h)$	(0.25, 8.01)	0.90	0.77	24.17	(27.87)	285.25	(458.91)
EWMA	$(r, L)$	(1.00, 3.090)	0.58	0.85	34.55	(38.94)	701.96	(1068.50)
EWMA	$(r, L)$	(0.75, 3.087)	0.64	0.83	32.08	(38.23)	654.44	(1000.28)
EWMA	$(r, L)$	(0.50, 3.071)	0.71	0.82	28.92	(35.42)	615.59	(969.71)
EWMA	$(r, L)$	(0.40, 3.054)	0.75	0.82	27.90	(34.74)	602.86	(948.69)
EWMA	$(r, L)$	(0.30, 3.023)	0.79	0.81	26.16	(32.86)	559.71	(892.78)
EWMA	$(r, L)$	(0.25, 2.998)	0.81	0.81	26.25	(33.19)	553.58	(884.68)
EWMA	$(r, L)$	(0.20, 2.962)	0.83	0.80	25.36	(32.20)	506.41	(847.37)
EWMA	$(r, L)$	(0.10, 2.814)	0.87	0.78	24.17	(30.38)	420.15	(702.72)
EWMA	$(r, L)$	(0.05, 2.615)	0.98	0.37	4.90	(13.62)	333.58	(566.51)
EWMA	$(r, L)$	(0.03, 2.437)	0.99	0.35	1.93	(7.13)	304.86	(490.30)
JSA	$(\hat{\delta}_{\min}, \beta, \gamma, \zeta)$	(0.500, 0.20, 1.50, 6.06)	0.12	0.94	27.57	(31.33)	489.99	(642.11)
JSA	$(\hat{\delta}_{\min}, \beta, \gamma, \zeta)$	(0.500, 0.20, 2.50, 4.63)	0.13	0.87	28.09	(33.04)	222.09	(265.45)
JSA	$(\hat{\delta}_{\min}, \beta, \gamma, \zeta)$	(1.000, 0.30, 1.50, 5.05)	0.12	0.88	26.29	(33.10)	236.44	(268.91)
JSA	$(\hat{\delta}_{\min}, \beta, \gamma, \zeta)$	(1.000, 0.30, 3.00, 4.39)	0.13	0.83	26.38	(33.64)	152.14	(166.09)
AFF	$\alpha$	0.005	0.86	0.79	27.12	(32.05)	819.36	(1162.97)
AFF	$\alpha$	0.008	0.87	0.73	25.78	(30.97)	577.88	(866.34)
AFF	$\alpha$	0.010	0.88	0.71	24.96	(30.10)	495.66	(760.56)

**Table 2** The detection performance for CUSUM, EWMA, JSA and AFF for a stream of over 750,000 observations with approximately 5000 changes of size  $\delta$ , with  $\delta \in \{0.25, 0.5, 1, 3\}$ , with all methods using a burn-in of length  $B = 50$  to estimate the in-control mean and variance. The parameter values used for CUSUM and EWMA are the recommended choices in Hawkins (1993) and Lucas and Saccucci (1990), respectively, yet there is a large variation in performance. The values for the JSA control parameters are a selection of those recommended in Jiang et al (2008). For AFF, we only need to vary the single parameter  $\alpha$ . The value of the step size for AFF used in this table is  $\eta = 0.01$ , with similar results for other values of  $\eta$ . The performance measures CCD, the proportion of changepoints correctly detected, and DNF, the proportion of detected changepoints that are not false detections, are defined in Section 2.2.2. The Average Run Lengths, ARL0 and ARL1, are defined in Section 2.2.1, while the SDRL0 and SDRL1 are their standard deviations, respectively.

detectors, resulting in an increase in false detections, etc.

Note that the results for the JSA method suggest it may not be suitable for a streaming data context. A particular concern is the low CCD value, which indicates that many true changepoints are missed. Further results (for more parameter choices) for JSA are provided in Section 5 in the Supplementary Material.

Additionally, Section 5 in the Supplementary Material provides further results for gamma-distributed data to evaluate how these methods perform when the normality assumption does not hold. As discussed in detail in Section 5.2 (Supplementary Material), the AFF method appears to perform well despite this model misspecification, in comparison to CUSUM, EWMA and JSA.

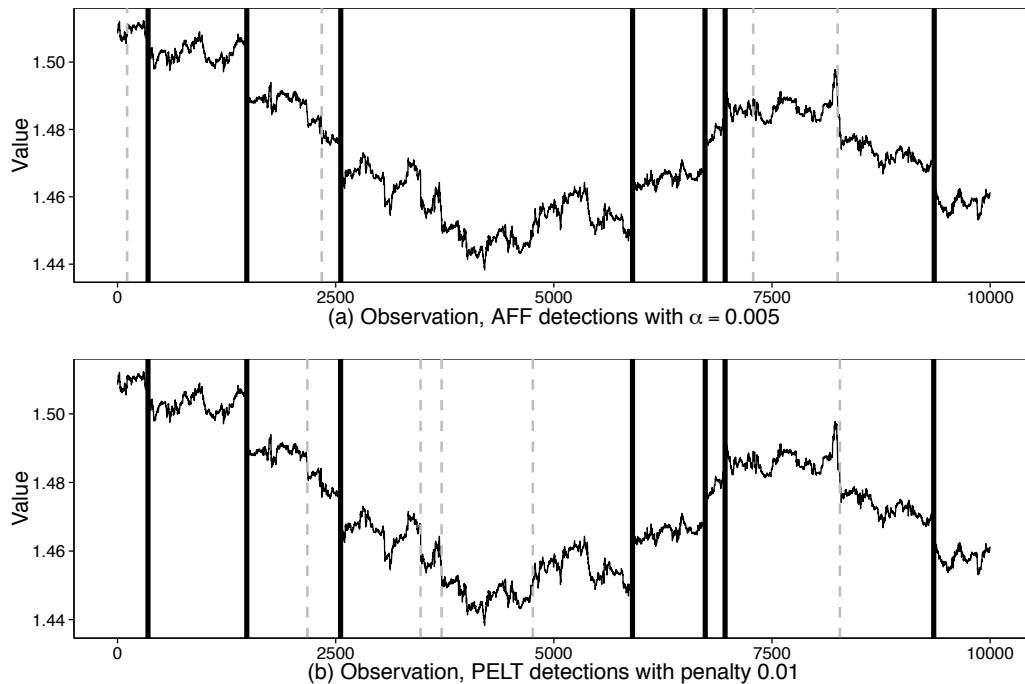
Note that while the SDRL0 and SDRL1 values may seem high, they are in line with the SDRL magnitudes in the literature (e.g. Jones et al 2001). These standard deviations are further inflated due to the estimation of the stream's mean and variance during the burn-in period, but it is not a large effect. This is discussed in more detail in the Supplementary Material (Sec. 3.4).

In the next section we investigate the performance of the AFF change detector on a real-world dataset.

## 6 Foreign exchange data

As discussed in the introduction, a primary application of continuous monitoring for data streams arises in financial trading. Here, the value of a financial instrument evolves over time, as a result of the behaviour of the market. Individual traders need to determine if the price has made an unexpected change in order to trigger trading actions. However, the data stream continues, uninterrupted, as such trading decisions happen. For illustration, we will consider 5-minute Foreign Exchange (FX) tick data. Specifically, we consider a stream of Swiss Franc (CHF) and Pound Sterling (GBP). Our objective here is simply to detect changes in the price-ratio, that could be used to trigger trading actions.

It is well known that FX streams are non-stationary. The standard approach to address this problem is to transform the data, and analyse the so-called log-returns,  $LR_t = \log(x_t) - \log(x_{t-1})$ .



**Fig. 5** Change detection on a CHF/GBP data stream using (a) AFF (above) and (b) PELT (below). The raw data stream is plotted with the detected changepoints indicated by the vertical lines, with solid black lines indicating that both schemes detect that changepoint (within 10 observations of each other), and grey dashed lines indicating that the changepoint is not detected by the other method. Note that the changepoint(s) around 8200 differs between PELT and AFF by 26 observations, and so is a grey dashed line.

We use AFF to perform change detection on the log-returns of the CHF/GBP data. There are over 330,000 observations. The data is from 07h05 on October 21, 2002 until 12h00 on May 15, 2007, with one data point every five minutes. For purposes of clarity, Figure 5(a) shows the changepoints (vertical lines) detected on the log-returns superimposed on the raw data stream for AFF with  $\alpha = 0.005$ .

Although this section is simply meant to provide an example of the AFF algorithm deployed on real data, we also provide a comparison with PELT (Killick et al 2012), an optimal *offline* detection algorithm, in order to provide an indication of the “true” changepoints. The changes detected by PELT is shown in Figure 5(b). PELT also has a single control parameter, which was chosen to be 0.01 for this figure. Note that PELT analysed the raw data stream, rather than the log-returns.

Figure 5 shows the first the changepoints detected on the first 10,000 observations for clarity, and shows a high degree of agreement between AFF and PELT; the two methods are declared to agree on a changepoint if they each detect within 10 observations of each other. For example, PELT detects a changepoint at observation 349, while AFF detects a changepoint at observation 350. In Figure 5, changepoints common to both methods are indicated by solid black lines, while

changepoints that are not detected by both methods are indicated by dashed grey lines.

On the full data set of over 330,000 observations, PELT detects 373 changepoints. AFF detects 154 of these 373 changepoints (within 10 observations of the PELT-detected changepoint), indicating that it detects over 40% of the changepoints detected by PELT. This is a particularly striking result, since AFF is an online method while PELT is an *offline* method<sup>2</sup>. Note however, that only 20% of the changepoints that AFF detects are changepoints also detected by PELT; in other words, AFF detects a large proportion of changepoints not detected by PELT. Similar behaviour is observed for both methods as parameter values are varied to increase sensitivity and to allow more changepoints to be detected. Section 7 in the Supplementary Material also compares the performance of CUSUM and EWMA with AFF on this data set. We use the R implementation of PELT provided in the `changepoint` package (Killick and Eckley 2011).

<sup>2</sup> Note that since it is an offline method it does not make sense to compute performance measures such as ARL0 and ARL1 for comparison with AFF, CUSUM and EWMA.

## 7 Conclusion

Continuous monitoring is much more challenging than the sequential detection of a single changepoint. Features that contribute to this challenge include defining different types of detections, restarting, determination of control parameters, and handling changes of different size. We have shown via simulation, in the simple context of detection of changes in the mean, that adaptive estimation performs similarly to CUSUM and EWMA, with a reduced burden on the analyst since only a single control parameter needs to be chosen. Moreover, to the best of our knowledge, this is the first performance analysis of CUSUM and EWMA on a datastream with multiple changes of different sizes.

The goal of developing this methodology was to obtain a method that is computationally efficient and can be confidently deployed on a real-world stream, without concern for selecting control parameters, and the AFF change detector satisfactorily meets this criteria.

Although we have only focused on monitoring the mean, the adaptive estimation framework can be extended to monitor the variance, or other parameters, of a stream. Additionally, it can be extended to a multivariate setting, which may arise, for example, in the Foreign Exchange problem described in Section 6 when handling multiple currency pairs.

## Acknowledgements

The work of Dean Bodenham was fully supported by a Roth Studentship provided by the Department of Mathematics, Imperial College, London. We are grateful to Christoforos Anagnostopoulos, David Hand, Nicholas Heard and Gordon Ross for helpful comments on this paper. All figures were created in R using the `ggplot2` package (Wickham 2009). Finally, we note that an R package `ffstream` implementing the AFF is in preparation.

## References

- Adams NM, Tasoulis DK, Anagnostopoulos C, Hand DJ (2010) Temporally-adaptive linear classification for handling population drift in credit scoring. In: Lechevallier Y, Saporta G (eds) COMPSTAT2010, Proceedings of the 19th International Conference on Computational Statistics, Springer, pp 167–176
- Aggarwal CC (ed) (2006) *Data Streams: Models and Algorithms*. Springer
- Anagnostopoulos C (2010) A statistical framework for streaming data analysis. PhD thesis, Imperial College London
- Anagnostopoulos C, Tasoulis DK, Adams NM, Pavlidis NG, Hand DJ (2012) Online linear and quadratic discriminant analysis with adaptive forgetting for streaming classification. *Statistical Analysis and Data Mining* 5(2):139–166
- Apley DW, Chin CH (2007) An optimal filter design approach to statistical process control. *Journal of Quality Technology* 39(2):93–117
- Appel U, Brandt AV (1983) Adaptive sequential segmentation of piecewise stationary time series. *Information Sciences* 29(1):27–56
- Åström K, Borisson U, Ljung L, Wittenmark B (1977) Theory and applications of self-tuning regulators. *Automatica* 13(5):457–476
- Åström KJ, Wittenmark B (1973) On self tuning regulators. *Automatica* 9(2):185–199
- Basseville M, Nikiforov IV (1993) *Detection of abrupt changes: theory and application*. Prentice Hall Englewood Cliffs
- Bodenham DA, Adams NM (2013) Continuous monitoring of a computer network using multivariate adaptive estimation. In: IEEE 13th International Conference on Data Mining Workshops (ICDMW), pp 311–388
- Bodenham DA, Adams NM (2014) Adaptive change detection for relay-like behaviour. In: IEEE Joint Information and Security Informatics Conference, 2014
- Borkar VS (2008) *Stochastic approximation: a dynamical systems viewpoint*. Cambridge University Press
- Capizzi G, Masarotto G (2003) An adaptive exponentially weighted moving average control chart. *Technometrics* 45(3):199–207
- Capizzi G, Masarotto G (2010) Self-starting CUS-CORE control charts for individual multivariate observations. *Journal of Quality Technology* 42(2):136–152
- Capizzi G, Masarotto G (2012) Adaptive generalized likelihood ratio control charts for detecting unknown patterned mean shifts. *Journal of Quality Technology* 44(4):281–303
- Choi SW, Martin EB, Morris AJ, Lee IB (2006) Adaptive multivariate statistical process control for monitoring time-varying processes. *Industrial & Engineering Chemistry Research* 45(9):3108–3118
- Fortescue T, Kershenbaum L, Ydstie B (1981) Implementation of self-tuning regulators with variable forgetting factors. *Automatica* 17(6):831–835
- Fraker SE, Woodall WH, Mousavi S (2008) Performance metrics for surveillance schemes. *Quality Engineering* 20(4):451–464

- Frisén M (2003) Statistical surveillance. optimality and methods. *International Statistical Review* 71(2):403–434
- Gama J (2010) *Knowledge Discovery from Data Streams*. Chapman Hall
- German RR, Lee LM, Horan JM, Milstein RL, Perkowski CA, Waller MN (2001) Updated guidelines for evaluating public health surveillance systems. *Morbidity and Mortality Weekly Report* 50:1–35
- Gustafsson F (2000) *Adaptive filtering and change detection*. Wiley
- Hawkins DM (1987) Self-starting Cusum charts for location and scale. *Journal of the Royal Statistical Society, Series D* 36(4):299–316
- Hawkins DM (1993) Cumulative sum control charting: an underutilized SPC tool. *Quality Engineering* 5(3):463–477
- Hawkins DM, Qiu P, Chang WK (2003) The change-point model for statistical process control. *Journal of Quality Technology* 35(4):355–366
- Haykin S (2002) *Adaptive Filter Theory*, 2002. Prentice-Hall
- Jensen WA, Jones-Farmer LA, Champ CW, Woodall WH, et al (2006) Effects of parameter estimation on control chart properties: a literature review. *Journal of Quality Technology* 38(4):349–364
- Jiang W, Shu W, Apley DW (2008) Adaptive cusum procedures with ewma-based shift estimators. *IIE Transactions* 40(10):992–1003
- Jones LA (2002) The statistical design of EWMA control charts with estimated parameters. *Journal of Quality Technology* 34(3):277–288
- Jones LA, Champ CW, Rigdon SE (2001) The performance of exponentially weighted moving average charts with estimated parameters. *Technometrics* 43(2):156–167
- Jones LA, Champ CW, Rigdon SE (2004) The run length distribution of the CUSUM with estimated parameters. *Journal of Quality Technology* 36(1):95–108
- Kalman RE (1960) A new approach to linear filtering and prediction problems. *Journal of Basic Engineering* 82(1):35–45
- Kifer D, Ben-David S, Gehrke J (2004) Detecting change in data streams. In: *Proceedings of the Thirtieth international conference on Very large data bases—Volume 30, VLDB Endowment*, pp 180–191
- Killick R, Eckley IA (2011) *Changepoint: an R package for changepoint analysis*. Lancaster University
- Killick R, Fearnhead P, Eckley IA (2012) Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association* 107(500):1590–1598
- Lorden G (1971) Procedures for reacting to a change in distribution. *The Annals of Mathematical Statistics* 1(6):1897–1908
- Lucas JM (1976) The design and use of V-mask control schemes. *Journal of Quality Technology* 8:1–11
- Lucas JM, Saccucci MS (1990) Exponentially weighted moving average control schemes: properties and enhancements. *Technometrics* 32(1):1–12
- Maboudou-Tchao EM, Hawkins DM (2013) Detecton of multiple change-points in multivariate data. *Journal of Applied Statistics* 40(9):1979–1995
- Moustakides GV (1986) Optimal stopping times for detecting changes in distributions. *The Annals of Statistics* 14(4):1379–1387
- Page E (1954) Continuous inspection schemes. *Biometrika* 41(1/2):100–115
- Pavlidis NG, Tasoulis DK, Adams NM, Hand DJ (2011) lambda-perceptron: an adaptive classifier for data streams. *Pattern Recognition* 44(1):78–96
- Roberts SW (1959) Control chart tests based on geometric moving averages. *Technometrics* 1(3):239–250
- Ross GJ, Adams NM, Tasoulis DK (2011) Nonparametric monitoring of data streams for changes in location and scale. *Technometrics* 53(4):379–389
- Sullivan JH (2002) Detection of multiple change points from clustering individual observations. *Journal of Quality Control* 34(4):371–383
- Tsung F, Wang T (2010) Adaptive charting techniques: literature review and extensions. In: Lenz H, et al (eds) *Frontiers in Statistical Quality Control*, 9, Springer-Verlag, pp 19–35
- Wickham H (2009) *ggplot2: elegant graphics for data analysis*. Springer New York
- Xie Y, Sigmund D (2013) Sequential multi-sensor change-point detection. *Annals of Statistics* 41(2):670–692