Parameter Estimation in Abruptly Changing Dynamic Environments Using Stochastic Learning Weak Estimator

Hugo Lewi Hammer[†][§] and Anis Yazidi[†]

Abstract

Many real-life dynamical systems experience abrupt changes followed by almost stationary periods. In this paper, we consider streams of data exhibiting such abrupt behavior and investigate the problem of tracking their statistical properties in an online manner.

We devise a tracking procedure where an estimator that is suitable for a stationary environment is combined together with an estimator suitable for a dynamic environment. The current estimate is based on the stationary estimator unless a statistically significant difference is observed between both estimators. The stationary estimate is deemed off track and a large update (jump) is given to get the stationary estimate back on track.

We use the Stochastic Learning Weak Estimator (SLWE) as the dynamic estimator. The SLWE is known to be the state-of-the art solution to tracking the properties of non-stationary environments, due to its multiplicative update form. Therefore, the SLWE is a better choice to accompany a stationary estimator than the far more common sliding window based approach.

A theoretically well founded statistical testing procedure is developed to detect a significant difference between the stationary and dynamical estimators. Although our procedure bears similarities to the event detection procedure suggested by Ross *et al.* (2012) [38], it is rather well founded theoretically. First, Ross *et al.* ignore the uncertainty in the stationary estimator in the detection procedure. Second, the detection threshold is determined based on heuristics and therefore lacks a solid statistical foundation.

Extensive simulation results, based on both synthetic and real-life data related to news topic classification, demonstrate that our estimation procedure is easy to tune and outperforms legacy works.

1 Introduction

The Maximum Likelihood Estimates (MLE) as well as the Bayesian estimation families operate with the premise that the distribution of the data being estimated is stationary over time. Under such settings, the convergence to the true value of the parameter being estimated takes place with probability 1 when the number of samples tends to infinity. However, in many real-life applications, the assumption on the stationarity of the data does not hold and the true underlying parameter being estimated changes over time. The Stochastic Learning Weak Estimators (SLWE) are known to be the state-of-the-art approach for such an estimation problem [32, 50]. The SLWE enjoys a multiplicative update form that makes it superior to the state-of-the-art estimation approaches which are mainly of additive flavor. However, the right choice of the intrinsic parameter of the SLWE, λ, is still an open issue. The latter parameter controls the forgetting of old data and controls the ability of the scheme to adapt to changes in the environments. If the system changes rapidly the parameter should be chosen to rapidly forget the old stale data. On the other hand, if the environment is stabilizing, the rate of forgetting should decrease.

The SLWE has found numerous successful applications in the literature. Applications of the SLWE include adaptive classifiers for spam filtering [50], adaptive file encoding with nonstationary distributions [40], intrusion detection in computer networks [44], tracking shifts of languages in online discussions [43], learning user preferences under concept-shift [31, 47], fault-tolerant

[†] OsloMet - Oslo Metropolitan University

[§] Corresponding author. Email: `hugo.hammer@hioa.no`

routing in Ad-hoc networks [30], digital content forensics for detecting illicit images [14], detection and tracking of malicious nodes in both Ad-hoc networks [34], vehicular mobile WiMAX networks [27], and optimizing firewall matching time via dynamic rule ordering [28]– to mention a few.

In many of such practical problems the dynamical system changes abruptly followed by periods where the system is almost stationary. Unfortunately, moving average estimators and SLWE are not well suited for such cases. By choosing a small moving average window or a high value of λ, the estimators will rapidly adjust after an abrupt change, but on the other hand, it will result in a higher estimation uncertainty when the system stabilizes. By choosing a large moving average window or a low value of λ, the estimation uncertainty will be low in the stationary parts, but on the other hand, the estimation procedures will suffer from adjusting too slowly after an abrupt change.

In this paper, we suggest a computationally efficient procedure to track statistical properties of environments that contain both abrupt changes and stationary parts. The procedure is based on running an estimator that is suitable for the stationary parts and an SLWE estimator that efficiently tracks changes in the data stream. In each iteration a novel and lightweight hypothesis testing mechanisms is performed to look for significant differences between the stationary and dynamic estimators. The current estimate is based on the stationary estimate except if a significant difference is detected, in which case the current estimate jumps to a new and more suitable value based on the current dynamic estimate. The suggested approach has similarities to the Exponentially Weighted Moving Average Control Scheme (EWMACS) [25] and Ross *et al.* [38] suggest to use the EWMACS for event detection. However, compared to our suggested approach, the approach by Ross *et al.* [38] has some clear weaknesses. First, the detection procedure is unreliable since its does not accommodate the uncertainty in the estimate of the mean value by the sample mean. As the uncertainty of the sample mean decreases with time, the detection threshold should be reduced with time. Thus the procedure will detect events too often (rare) for small (large) sample sizes. Second, the detection threshold is computed based on some heuristics that can not be related to a statistical significance quantity. Therefore, it is difficult to determine a suitable value for the detection threshold. Finally please note that the focus in [38] is different from ours. [38] focuses on detecting concept drift in a supervised learning setting (change in the relations between features and response) while our focus is tracking statistical properties of data streams. We present the estimation procedure for the binomial and multinomial distributions, but can be applied to other distributions as well.

The article is organized as follows. In Section 2 we review the related research. In order to make the article self-contained, a brief description of the SLWE estimator is given in Section 3. Section 4 gives the details of our approach. In Section 5 we extend the scheme to the multinomial case. In Section 6 we perform thorough evaluation of the algorithms and draw some final conclusions in Section 7.

2 Related Work and State-of-the-Art

In this Section we review the related work. First, in Section 2.1 we will review legacy scheme for estimation under non-stationary environment. Then, in Section 2.2 we will review the different approaches for controlling the parameters of estimators operating in non-stationary environments. Finally, in Section 2.3 we review tracking algorithms suitable for environments that contain both abrupt changes and stationary parts.

2.1 Estimation in Non-Stationary Environments

Probably, the most classical and utilized method for dealing with non-stationary estimation problems is the *sliding window* approach which can be seen as a short memory version of the MLE. According to the sliding window approach, the last samples that fit in the window are used to compute the estimates online. Nevertheless, the *sliding window* method suffers from a tuning problem. In fact, if the size of the window is chosen too large, then the quality of the estimates will be deteriorated by stale data values, while choosing a too small window size would rather lead to poor estimates with low confidence.

A myriad of works have been proposed to address detecting change points. Those methods fall under two main families: Page's cumulative sum (CUSUM) [2] detection procedure, and the Shiryaev-Roberts-Pollak detection procedure. In [42], Shiryayev resorted to a Bayesian formulation in which the change point is assumed to have a geometric prior distribution. CUSUM uses the idea of maximum likelihood ratio test hypothesis to discern change points. However, a downside of these two approaches is their computational complexity which renders the SLWE as well as the estimator in this paper lightweight alternatives.

When it comes to extensions of the sliding window, Koychev *et al.* proposed a new paradigm called Gradual Forgetting (GF) [19, 21, 22]. According to the principles of GF, observations in the same window are treated unequally when computing the estimates based on weight assignment. Recent observations receive more weights than distant ones. Different forgetting functions were proposed ranging from linear [20] to exponential [18].

In [32], Oommen and Rueda presented the SLWE to estimate the underlying parameters of time varying binomial/multinomial distribution. The update form of the SLWE is inspired from the theory of variable structure Learning Automata [29], and more particularly, its reward-inaction flavor. The most appealing properties of the SLWE which makes it the state-of-the-art is its multiplicative form of updates. Two different counter-parts of SLWE [32] for discretized spaces was recently proposed in [49] and [48]. In a similar manner to the SLWE, the latter solution also suffers from the problem of tuning the resolution parameter.

2.2 Estimation using Adjustable parameters

In this Section, we survey some of the most pertinent techniques for estimation in dynamic environments that are orthogonal to the SLWE. For a thorough survey we refer the reader to the surveys [10, 23] which provide a comprehensive taxonomy of estimation methods in non-stationary environments, namely, *adaptive windowing*, *aging factors*, *instance selection* and *instance weighting.*

Gama *et al.* [10] presents a clear distinction between memory management and forgetting mechanisms. Adaptive windowing [46] works with the premise of growing the size the sliding window indefinitely until a change is detected via a change detection technique. In this situation, the size of the window is reset whenever a changed is detected.

Another interesting family of approaches assume that the true value of the parameter being estimated is revealed after some delay, which enables quantifying the error of the estimator. In such settings, some research [45] have used ensemble methods where the output of different estimators is combined using weighted majority voting. The weights of each estimator is adjusted based on its error. In this sense, estimation methods that produce high error see their weight decrease.

In the same perspective, the estimated error can be used for re-initializing the estimation as performed in [38]. In all brevity, changes are detected based on comparing sections of data, using statistical analysis to detect distributional changes, i.e., abrupt or gradual changes in the mean of the data points when compared with a baseline mean with a random noise component. One option is also to keep a reference window and compare recent windows with the reference window to detect changes [8]. This can, for example, be done based on comparing the probability distributions of the reference window and the recent window using Kullback-Leibler divergence [6, 41].

2.3 Estimation in Environments With Both Stationary Parts and Abrupt Changes

As described above there are several algorithms to track statistical properties of dynamically varying environments and to detect changes. However, procedures specifically constructed to track statistical properties of environments with both stationary parts and abrupt changes are very sparse. A prominent exception is the ADWIN2 procedure [5]. ADWIN2 keeps a variable-length window of recently seen items, with the property that the window has the maximal length statistically consistent with the hypothesis there has been no change in the average value inside the window. More precisely, an older fragment of the window is dropped if and only if there is enough evidence that its average value differs from that of the rest of the window. This makes the procedure suitable for tracking statistical properties of environments with both stationary parts and abrupt changes.

Another common approach is to combine a tracking procedures, as described in Sections 2.1 and 2.2 with a change detection procedure. When a change is detected, the memory is flashed and the tracking restarts [17]. A representative set of approaches that fall under the latter category include [39, 37, 36, 11, 12, 13]. For instance, in [39], Ross *et al.* addressed the problem of change detection of Bernoulli random variables using the Fishers Exact Test (FET). The assumption is that the pre-change value of the Bernoulli parameter is unknown in advance. Hawkins and his collaborators used the Student-t, Bartlett and Generalized Likelihood Ratio (GLR) methods for detecting changes in mean and variance in a sequence of Gaussian random variables in a series of works [11, 12, 13]. The Mann-Whitney and Lepage test statistics are used in [37] with no assumptions made about the distribution of the observations.

3 Stochastic Learning Weak Estimator

Let $X_1, X_2, X_3, \ldots$ represent a stream of independent and identically distributed Bernoulli stochastic variables with parameter p. That is

$$\begin{aligned} P(X_n = 0) &= 1 - p \\ P(X_n = 1) &= p \end{aligned} \tag{1}$$

for $n = 1, 2, 3, \ldots$.

We now want to estimate the parameter p from the stream of Bernoulli variables. Using the weak estimator, the estimate of p is updated by the following recursion

$$\begin{aligned} \hat{p}_1 &= X_1 & \\ \hat{p}_n &= \lambda_n \hat{p}_{n-1} & \text{if } X_n = 0 \\ \hat{p}_n &= 1 - \lambda_n(1 - \hat{p}_{n-1}) & \text{if } X_n = 1 \end{aligned} \tag{2}$$

where $\hat{p}_n$ represents the estimate of p after the arrival of X_n and $\lambda_n,\ n = 1, 2, \ldots$ are constants between zero and one. The intuition is that if $X_n = 0$ we should reduce our current estimate of p (the probability of one) which is achieved by multiplying the current estimate of p by λ_n. On the other hand, if $X_n = 1$ we should reduce the estimate of $1 - p$ (the probability of zero) which gives

$$\begin{aligned} 1 - \hat{p}_n &= \lambda_n(1 - \hat{p}_{n-1}) \\ \hat{p}_n &= 1 - \lambda_n(1 - \hat{p}_{n-1}) \end{aligned}$$

which is equal to the last equation in (2).

The recursions in (2) can be written as follows

$$\hat{p}_n = X_n(1 - \lambda_n(1 - \hat{p}_{n-1})) + (1 - X_n)\lambda_n \hat{p}_{n-1},\ n = 1, 2, \ldots$$

with $\lambda_1 = 0$. Using straight forward calculations this simplifies to

$$\hat{p}_n = \lambda_n \hat{p}_{n-1} + (1 - \lambda_n) X_n \tag{3}$$

which can be recognized as the exponentially weighted moving average.

We can prove by induction that $\hat{p}_n$ is an unbiased estimator for p for every n as follows

$$\begin{aligned} E(\hat{p}_1) &= E(X_1) = p \quad \text{(recall } \lambda_1 \text{ is set to 0)} \\ E(\hat{p}_n) &= E(\lambda_n \hat{p}_{n-1} + (1 - \lambda_n) X_n) \\ &= \lambda_n p + (1 - \lambda_n) p \\ &= p \end{aligned}$$

The variance depends on the choice of the λ's. We look at two special cases.
λ constant: It can be proved that if we set all $\lambda_n = \lambda$, the limiting variance is given by [50]

$$\lim_{n \to \infty} \operatorname{Var}(\hat{p}_n) = \frac{1 - \lambda}{1 + \lambda} p(1 - p)$$

An advantage of the constant λ approach is that if the value of p is changing with time in the underlying Bernoulli data stream, the estimator will rapidly adjust to these changes [50]. A disadvantage is that if p is not changing, the variance of the estimator will have a lower limit and never reaches zero.

Sample mean: The sample mean is the maximum likelihood estimator of p and is the natural estimator to use if p is not changing with time. Let $\bar{p}_{n-1}$ denote the sample mean of the first $n - 1$ Bernoulli variables from the stream

$$\bar{p}_{n-1} = \frac{1}{n - 1} \sum_{i=1}^{n-1} X_i$$

When X_n arrives, the sample mean can be updated as follows

$$\bar{p}_n = \frac{n - 1}{n} \bar{p}_{n-1} + \frac{1}{n} X_n \tag{4}$$

which is equivalent to (3) with $\lambda_n = (n - 1)/n$. This means that the sample mean is a special case of the general recursion in (3). It is well known that $\lim_{n \to \infty} \operatorname{Var}(\bar{p}_n) = 0$. A disadvantage of the sample mean is that if p is changing with time, the sample mean will become very slow at adjusting to these changes. On the other hand if p is not changing, the sample mean is the optimal estimator in the sense that no other unbiased estimators can achieve less variance.

4 Estimation in a shifting environment

Suppose a situation where p is switching between different values with time. An example could be a news stream where the topic of the news stream suddenly changes due to different real life events. Another example could be a machine operated by different employees at different time periods each with its own error rate characterized by p. We assume that the instants in which p switches values are unknown.

For such systems a possible strategy would be to use the sample mean whenever the p is not changing, and a mechanism to "jump" fast towards a new estimate if the value of p has changed. In this paper we suggest a method that combines the sample mean and a weak estimator with constant λ. Let $\hat{p}_n^\lambda$ and $\bar{p}_n$ denote the weak estimator with constant λ and the sample mean, respectively, after the arrival of X_n. If p switches value, $\hat{p}_n^\lambda$ will rapidly adjust to the new value of p, while minor changes will appear to $\bar{p}_n$. This can be used to build an efficient method to detect changes in p and "jump" to the new value of p.

Ross *et al.* [38] suggest a detection procedure based on the EWMACS [25]. However, the procedure has some clear weaknesses compared to the approach suggested in this paper. First, [38] focuses only on the uncertainty in $\hat{p}_n^\lambda$ and thus completely ignores the uncertainty in $\bar{p}_n$ making the procedure unreliable. For example, since the uncertainty of $\bar{p}_n$ decreases with n, the threshold should be reduced with time and is not taken into account in [38]. Thus the procedure will detect events too often for small values of n compared to for larger values of n. Secondly, the detection procedure is not within a statistical framework, making it difficult to find a suitable detection threshold value since it cannot be connected to a statistical significance value. Finally please note that the focus in [38] is different from ours. [38] focuses on detecting concept drift in a supervised learning setting (change in the relations between features and response) while our focus is on tracking statistical properties of the data stream.

In this paper we will build a novel detection procedure based on analyzing the distribution of the difference between the estimators, namely $\hat{p}_n^\lambda - \bar{p}_n$. Developing a statistical test based on the difference $\hat{p}_n^\lambda - \bar{p}_n$ is an example of the *pivotal quantity* method and probably is the most common method to develop statistical tests [4]. The method gives good control over the uncertainty in the testing procedure. If p switches value we expect that $\hat{p}_n^\lambda - \bar{p}_n$ will be large in absolute value and larger then what we would expect if p remains constant. This can be used to build a statistical test if p has changed value or not. Bifet *et al.* [5] build tests in a similar manner in ADWIN2 based on comparing windows of observations, but the mathematics and statistical analyzes will be slightly more involved for the SLWE case as considered in this paper.

We start by presenting the expectation and variance of the difference $\hat{p}_n^\lambda - \bar{p}_n$.

Theorem 1. *Let $X_1, X_2, X_3, \ldots$ represent a stream of independent and identically distributed Bernoulli stochastic variables with parameter p. Further let $\hat{p}_n^\lambda$ and $\bar{p}_n$ denote the weak estimator with constant λ and the sample mean, respectively, after the arrival of X_n. Then*

$$E\left(\hat{p}_n^\lambda - \bar{p}_n\right) = 0 \tag{5}$$

$$Var\left(\hat{p}_n^\lambda - \bar{p}_n\right) = p(1-p)\left(\left(\frac{1}{n} - \lambda^{n-1}\right)^2 + \sum_{i=2}^{n}\left(\frac{1}{n} - (1-\lambda)\lambda^{n-i}\right)^2\right) \tag{6}$$

Proof. We start by computing through the recursions in (3)

$$\begin{aligned}
\hat{p}_1 &= X_1 \\
\hat{p}_2 &= \lambda_2\hat{p_1} + (1-\lambda_2)X_2 \\
&= \lambda_2 X_1 + (1-\lambda_2)X_2 \\
\hat{p}_3 &= \lambda_3\hat{p_2} + (1-\lambda_3)X_3 \\
&= \lambda_3[\lambda_2 X_1 + (1-\lambda_2)X_2] + (1-\lambda_3)X_3 \\
&= \lambda_3\lambda_2 X_1 + \lambda_3(1-\lambda_2)X_2 + (1-\lambda_3)X_3 \\
\vdots \quad & \quad\quad \vdots \\
\hat{p}_n &= \sum_{i=1}^{n} X_i(1-\lambda_i)\prod_{j=i+1}^{n}\lambda_j, \text{ with } \lambda_1 = 0
\end{aligned}$$

Setting $\lambda_n = \lambda$ (and still $\lambda_1 = 0$) we get

$$\hat{p}_n = X_1\lambda^{n-1} + \sum_{i=2}^{n} X_i(1-\lambda)\lambda^{n-i}$$

and setting $\lambda_n = (n-1)/n$ we get the sample mean $\hat{p}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$.

Now we are ready to compute the expectation and variance

$$\begin{aligned}
E(\hat{p}_n^\lambda - \bar{p}_n) &= E\left(\frac{1}{n}\sum_{i=1}^n X_i - X_1\lambda^{n-1} - \sum_{i=2}^n X_i(1-\lambda)\lambda^{n-i}\right) \\
&= \frac{1}{n}\sum_{i=1}^n p - p\lambda^{n-1} - p\sum_{i=2}^n (1-\lambda)\lambda^{n-i} = \\
&= p - p\lambda^{n-1} - p(1-\lambda)\sum_{i=0}^{n-2}\lambda^i \\
&= p - p\lambda^{n-1} - p(1-\lambda)\frac{1-\lambda^{n-1}}{1-\lambda} \\
&= 0
\end{aligned}$$

$$\begin{aligned}
\mathrm{Var}(\hat{p}_n^\lambda - \bar{p}_n) &= \mathrm{Var}\left(\frac{1}{n}\sum_{i=1}^n X_i - X_1\lambda^{n-1} - \sum_{i=2}^n X_i(1-\lambda)\lambda^{n-i}\right) = \\
&= \mathrm{Var}\left(\left(\frac{1}{n} - \lambda^{n-1}\right)X_1 + \sum_{i=2}^n \left(\frac{1}{n} - (1-\lambda)\lambda^{n-i}\right)X_i\right) \\
&= p(1-p)\left(\left(\frac{1}{n} - \lambda^{n-1}\right)^2 + \sum_{i=2}^n \left(\frac{1}{n} - (1-\lambda)\lambda^{n-i}\right)^2\right)
\end{aligned}$$

□

Please note that $\mathrm{Var}(\hat{p}_n^\lambda - \bar{p}_n)$ can be computed recursively such that all variances up to n can be computed in $O(n)$ time. The actual recursions are not shown, but are straightforward to compute from (6). Another appealing property is that the variance does not depend on the stream of observations and can be computed before the data streaming starts. This lays the foundations for building very efficient algorithms.

Theorem 1 stated the expectation and variance of the distribution of $\hat{p}_n^\lambda - \bar{p}_n$. Next we investigate other properties the distribution. From the proof we saw that $\hat{p}_n^\lambda - \bar{p}_n$ could be written as follows

$$\hat{p}_n^\lambda - \bar{p}_n = \left(\frac{1}{n} - \lambda^{n-1}\right)X_1 + \sum_{i=2}^n \left(\frac{1}{n} - (1-\lambda)\lambda^{n-i}\right)X_i$$

which is a weighted sum of the independent Bernoulli variables. If the sum satisfies the Lindeberg criterion (and thus the Lyapunov criterion), the sum will, according to the central limit Theorem, converge to a normal distribution [16]. Unfortunately the sum does not satisfy this criterion (proofs omitted). A second option is to study the distribution of $\hat{p}_n^\lambda - \bar{p}_n$ by stochastic simulation. We perform the following experiment. We generated $n = 50$ independent outcomes from the Bernoulli distribution and computed $\hat{p}_{50}^\lambda - \bar{p}_{50}$ using $\lambda = 0.95$. Further we repeated this procedure $N = 10\,000$ times. The upper left panel in Figure 1 shows the histogram of these values when $p = 0.1$. The black curve is the normal distribution with expectation and variance as given by Theorem 1. The upper right panel shows the same, but with $n = 1000$. The second and the third row shows the same but with $p = 0.3$ and $p = 0.5$. Overall we see that the distribution of $\hat{p}_n^\lambda - \bar{p}_n$ is almost identical to a normal distribution. We only observe that when p is small (or high) the distribution is a little asymmetric compared to a normal distribution. Based on these observations it is a reliable choice to build a test assuming that $\hat{p}_n^\lambda - \bar{p}_n$ is normally distributed. We then get the following test.

Theorem 2. *Let $X_1, X_2, X_3, \ldots$ represent a stream of independent and identically distributed Bernoulli stochastic variables with parameter p. Further let z_α denote the α quantile of the standard normal distribution. Define the hypotheses*

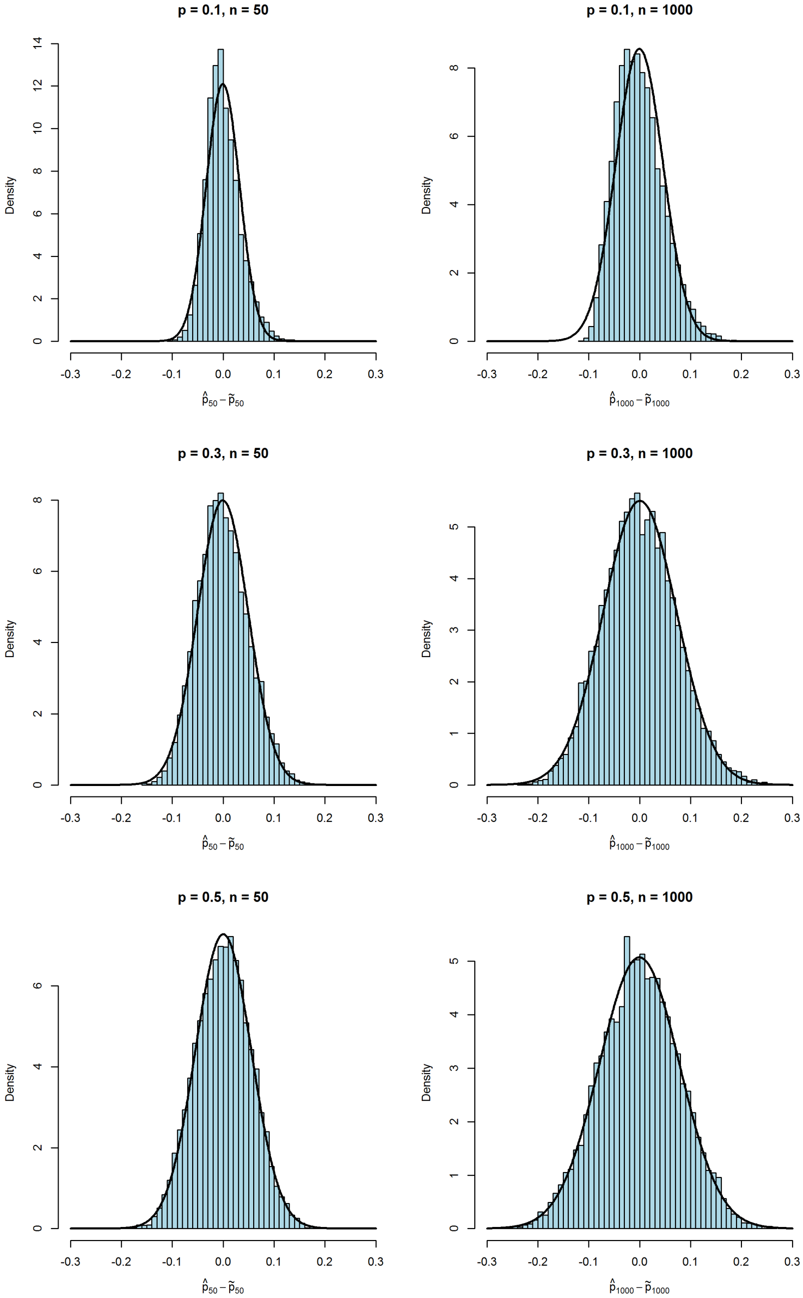

Figure 1: The distribution of $\hat{p}_n^\lambda - \bar{p}_n$ and the normal distribution with the same expectation and variance (black curve).

H_0: The underlying p has not changed value

H_1: The underlying p has changed value

Suppose that we decide to reject H_0 if

$$\frac{|\hat{p}_n^\lambda - \bar{p}_n|}{\sqrt{Var(\hat{p}_n^\lambda - \bar{p}_n)}} > z_{\alpha/2} \tag{7}$$

Then the probability of rejecting H_0 if H_0 is true is approximately α and the rejection rule (7) *controls the type I error.*

Proof. Let $N(\mu, \sigma)$ denote a normal distribution with expectation μ and standard deviation σ. From the discussion above and Figure 1 we know that

$$\frac{\hat{p}_n^\lambda - \bar{p}_n}{\sqrt{\mathrm{Var}(\hat{p}_n^\lambda - \bar{p}_n)}} \approx N(0,1)$$

which means that

$$\begin{aligned} P(\text{Type I error}) = P(\text{reject } H_0 \,|\, H_0 \text{ true}) = \\ = P\left(\frac{|\hat{p}_n^\lambda - \bar{p}_n|}{\sqrt{\mathrm{Var}(\hat{p}_n^\lambda - \bar{p}_n)}} > z_{\alpha/2}\right) \approx \alpha \end{aligned}$$

□

From Theorem 1 we see that $\mathrm{Var}(\hat{p}_n^\lambda - \bar{p}_n)$ depends on p which of course is unknown. To perform the test above, a natural choice is to substitute p with the sample mean $\bar{p}_n$ since this is our best estimate of p under the hypothesis that p is constant.

The basic idea of our method is to estimate p using the sample mean, but perform occasional jumps if the test in Theorem 2 brings evidence that p has switched value. In the next section we discuss different alternatives to perform the jumps.

4.1 Performing a jump

Let $\tilde{p}_n$ denote the estimate using the sample mean with jumps method after the arrival of X_n. Further let $\tilde{\lambda}_n$ denote the value used for λ_n in the recursions in (3) to compute $\tilde{p}_n$. Assume now that the test in Theorem 2 brings evidence that p has switched value which means that the current estimate $\tilde{p}_n$ is not reliable (since it is based on the sample mean). Therefore we need to adjust the estimate $\tilde{p}_n$ (jump). Two options seem natural to perform the jump.

- Forget the whole estimation history and set $\tilde{p}_n = X_n$
- Assume that the current estimate based on the weak estimator with constant λ, $\hat{p}_n^\lambda$, is reliable since it adjusts fast and set $\tilde{p}_n = \hat{p}_n^\lambda$.

A third option could be to set $\tilde{p}_n$ equal to some weighting between these two alternatives.

To continue the update of the estimator $\tilde{p}_n$ after the jump, we need to decide a new value for $\tilde{\lambda}_n$ as well. There are at least two natural alternatives

- Recall that with $\lambda_n = (n-1)/n$ in (3), we get the sample mean. If we decide to follow the first option above and set $\tilde{p}_n = X_n$, $\tilde{p}_n$ is just the sample mean of one observation which means that it is natural to set $\tilde{\lambda}_n = (1-1)/1 = 0$.

- If we decide to follow the second option above and set $\tilde{p}_n = \hat{p}_n^\lambda$, it seems natural to do the next update of $\tilde{p}_n$ similar to the update $\hat{p}_n^\lambda$, which means to relate $\tilde{\lambda}_n$ to λ. Since we will continue to update $\tilde{p}_n$ according to the sample mean, we must relate such a choice to the number of terms in a sample mean. We do this as follows. Define $\tilde{n}$ as the solution of the equation

$$\frac{\tilde{n}-1}{\tilde{n}} = \lambda$$

Solving with respect to $\tilde{n}$ and rounding off to the nearest integer we get

$$\tilde{n} = \left[\frac{1}{1-\lambda}\right]$$

where $[a]$ denotes the value of a a rounded of to the nearest integer. The interpretation of $\tilde{n}$ is the number of terms in a sample mean in which an update of the estimate will be similar to the weak estimator $\hat{p}_n^\lambda$.

Note that the choice of $\tilde{\lambda}_n$ in the first alternative above is equivalent to setting $\tilde{n} = 1$. It may be that when the test in Theorem 2 detects a change in p, the value of $\hat{p}_n^\lambda$ has not converged completely around the new value of p. Therefore a value of $\tilde{n}$ somewhere between 1 and $[1/(1-\lambda)]$ may be an even better alternative. By relating the variance of $\tilde{p}_n$ to a sample mean with $\tilde{n}$ terms, the variance $\mathrm{Var}(\hat{p}_n^\lambda - \tilde{p}_n)$, which we need in the test in Theorem 2, can be computed recursively. In addition, all the variances can recursively be computed in advance before the data stream starts.

Before the algorithm can be run, we need to decide a value for α in the test proposed in Theorem 2 as well. When we run the test, the probability of wrongly detecting a change in p, is approximately α. In practice we may run the test many times, for example every tenth iteration. If we run the test many times, the chance of wrongly detecting a change in p in some of these tests naturally will be larger then α. This refers to the multiple testing problem in the statistical literature, see e.g. [3]. A simple and much used approach is the Bonferroni correction where a significance level of α/M is used instead of α, where M is the number of tests. There are two challenges with applying this approach (and other standard corrections). First, we do not know the number of tests we need to run. Second, the Bonferroni correction assumes that all the tests are independent. In our case this is far from true, since two subsequent tests are based on almost the same data stream (only a few extra observation have been added since the last test) and the outcomes are highly correlated. Using the Bonferroni correction will result in a too low significance level, and the tests may never detect that p has changed. In practice, setting α to about 10^{-3} overall performs well and is, as expected, somewhere between standard significance levels (0.05) and Bonferroni corrected levels.

The algorithm using the second option above is shown in Algorithm 1.

5 Extension to the multinomial case

We now show how the jump algorithm above can be extended to the multinomial case. As described above, a Bernoulli variable takes the values 0 or 1 with probabilities $1-p$ and p, respectively. For the multinomial case this is extended such that X takes one of the values $\{1, 2, \ldots, r\}$ with probabilities $\{p_1, p_2, \ldots, p_r\}$, such that $\sum_{i=1}^r p_i = 1$. For ease of presentation below, define a stochastic vector Y which is a map from X as follows

$$Y = [\mathbb{I}(X=1), \mathbb{I}(X=2), \ldots, \mathbb{I}(X=r)] \tag{8}$$

where $\mathbb{I}(A)$ denote the indicator function returning one if A is true and zero if A is false. We see that Y is a vector with value one in position X and zero in all the other positions.

Algorithm 1 The sample mean with jumps algorithm.

Input:

```
$X_1, X_2, X_3, \dots$ //Stream of Bernoulli variables
$\lambda$
$\alpha$
$D$ //How often to perform the test in Theorem 2
$N$ //Max number of iterations
```

Method:

```
$\tilde{n} \leftarrow 0$
$\hat{p}_1^\lambda \leftarrow X_1$
$\tilde{p}_1 \leftarrow X_1$
for $n \in 1, 2, \dots, N$ do
    $\hat{p}_n^\lambda \leftarrow \lambda \hat{p}_{n-1}^\lambda + (1-\lambda) X_n$
    $\tilde{n} \leftarrow \tilde{n} + 1$
    $\tilde{p}_n \leftarrow \frac{\tilde{n}-1}{\tilde{n}} \tilde{p}_{n-1} + \frac{1}{\tilde{n}} X_n$
    if $n \bmod D == 0$ then
        if $\frac{|\hat{p}_n^\lambda - \tilde{p}_n|}{\sqrt{\mathrm{Var}(\hat{p}_n^\lambda - \tilde{p}_n)}} > z_{\alpha/2}$ then
            $\tilde{p}_n \leftarrow \hat{p}_n^\lambda$
            $\tilde{n} \leftarrow [1/(1-\lambda)]$
        end if
    end if
end for
```

Let $Y_1, Y_2, Y_3, \dots$ denote a stream of independent stochastic variables identical to Y. We now want to maintain running estimates of the probabilities $\{p_1, p_2, \dots, p_r\}$. The SLWE in (3) can easily be extend to the multinomial case as follows

$$[\hat{p}_{n,1}, \dots, \hat{p}_{n,r}] = \lambda_n [\hat{p}_{n-1,1}, \dots, \hat{p}_{n-1,r}] + (1-\lambda_n) Y_n \tag{9}$$

where $\hat{p}_{i,n}$ denote the estimate of p_i after receiving the variable Y_n from the data stream.

Now let $[\hat{p}_{n,1}^\lambda, \dots, \hat{p}_{n,r}^\lambda]$ denote estimates based on (9) using a constant value of λ and let $[\overline{p}_{n,1}, \dots, \overline{p}_{n,r}]$ denote the sample mean, i.e. using $\lambda_n = (n-1)/n$. Following the same argumentation as in Section 4 we know that

$$\frac{\hat{p}_{n,i}^\lambda - \bar{p}_{n,i}}{\sqrt{\mathrm{Var}(\hat{p}_{n,i}^\lambda - \bar{p}_{n,i})}} \approx N(0,1), \quad i = 1, 2, \dots, r \tag{10}$$

As an extension to Section 4, we now want to construct a statistical test to check wether the unknown probability vector $[p_1, p_2, \dots, p_r]$ has changed value. A common statistical test on the probability vector of the multinomial distribution is the Pearson's χ^2 test [1]. Adapting the χ^2 test to the application in this paper, we get the following theorem.

Theorem 3. *Let $Y_1, Y_2, Y_3, \dots$ represent a stream of independent and identically distributed multinomial stochastic vectors with probability vector $[p_1, p_2, \dots, p_r]$. Further let $\chi^2_{n,\alpha}$ denote the α quantile of the χ^2 distribution with n degrees of freedom. Define the hypotheses*

H_0: The underlying probability vector $[p_1, p_2, \dots, p_r]$ has not changed value

H_1: The underlying probability vector $[p_1, p_2, \dots, p_r]$ has changed value

Suppose that we decide to reject H_0 if

$$\sum_{i=1}^{r} \frac{\left(\hat{p}_{n,i}^\lambda - \bar{p}_{n,i}\right)^2}{Var(\hat{p}_{n,i}^\lambda - \bar{p}_{n,i})} > \chi^2_{r-1,\alpha} \tag{11}$$

Then the probability of rejecting H_0 if H_0 is true is approximately α and the rejection rule (11) *controls the type I error.*

Proof. It is well known that the sum of n independent squared standard normally distributed stochastic variables is χ^2_n distributed, denoting a χ^2 distribution with n degrees of freedom. From (10) we see that the sum in (11) is a sum of approximately squared standard normally distributed stochastic variables and therefore is approximately χ^2 distributed. Knowing $r-1$ terms in the sum, the last term can be computed since the probability estimates sum to one. The sum in (11) thus is approximately χ^2_{r-1} distributed

$$\sum_{i=1}^{r} \frac{\left(\hat{p}^{\lambda}_{n,i} - \bar{p}_{n,i}\right)^2}{\mathrm{Var}(\hat{p}^{\lambda}_{n,i} - \bar{p}_{n,i})} \approx \chi^2_{r-1} \tag{12}$$

Theorem 3 follows directly from (12). □

Algorithm 2 shows the resulting jump algorithm for the multinomial case.

Algorithm 2 The sample mean with jumps algorithm for the multinomial case.

Input:

$Y_1, Y_2, Y_3, \ldots$ //Stream of multinomial variables on vector form (recall Eq. (8))

λ

α

D //How often to perform the test in Theorem 2

N //Max number of iterations

Method:

```
$\tilde{n} \leftarrow 0$
$[\hat{p}^{\lambda}_{1,1}, \ldots, \hat{p}^{\lambda}_{1,r}] \leftarrow Y_1$
$[\tilde{p}_{1,1}, \ldots, \tilde{p}_{1,r}] \leftarrow Y_1$
for $n \in 1, 2, \ldots, N$ do
    $[\hat{p}^{\lambda}_{n,1}, \ldots, \hat{p}^{\lambda}_{n,r}] \leftarrow \lambda[\hat{p}^{\lambda}_{n,1}, \ldots, \hat{p}^{\lambda}_{n,r}] + (1-\lambda)Y_n$
    $\tilde{n} \leftarrow \tilde{n} + 1$
    $[\tilde{p}_{n,1}, \ldots, \tilde{p}_{n,r}] \leftarrow \frac{\tilde{n}-1}{\tilde{n}}[\tilde{p}_{n-1,1}, \ldots, \tilde{p}_{n-1,r}] + \frac{1}{\tilde{n}}Y_n$
    if $n \bmod D == 0$ then
        if $\sum_{i=1}^{r} \frac{\left(\hat{p}^{\lambda}_{n,i} - \tilde{p}_{n,i}\right)^2}{\mathrm{Var}(\hat{p}^{\lambda}_{n,i} - \tilde{p}_{n,i})} > \chi^2_{r-1,\alpha}$ then
            $[\tilde{p}_{n,1}, \ldots, \tilde{p}_{n,r}] \leftarrow [\hat{p}^{\lambda}_{n,1}, \ldots, \hat{p}^{\lambda}_{n,r}]$
            $\tilde{n} \leftarrow [1/(1-\lambda)]$
        end if
    end if
end for
```

6 Experiments

In this Section we evaluate the methodology above for both synthetic and real-life data. In all the experiments reported below, we set $\tilde{p}_n = \hat{p}^{\lambda}_n$ after a jump, i.e. the second alternative discussed in Section 4.1 and as given in Algorithms 1 and 2.

6.1 Synthetic data example

We will evaluate the binomial case (Algorithm 1) and the multinomial case for $r = 4$ classes (Algorithm 2). Figure 2 visualizes some tracking procedures for a binomial case with large changes in p. The gray, green and blue curves show the Algorithm 1 ($\tilde{p}_n$), the SLWE with

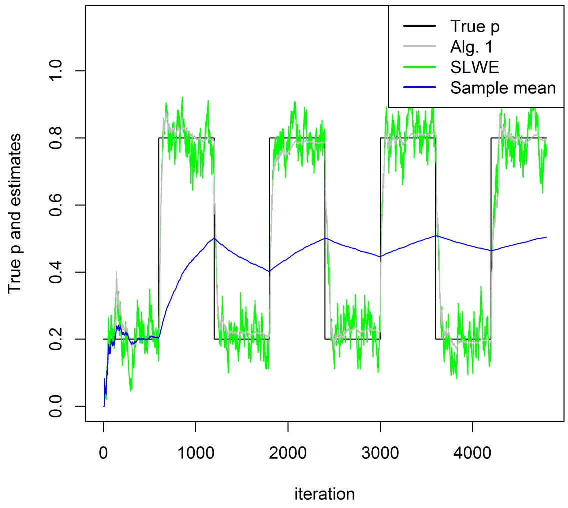

Figure 2: Evaluation of the estimators in an environment with large jumps in p. The black curve shows the true p in each iteration. The gray, green and blue curves show the estimators $\tilde{p}_n$, $\hat{p}_n^\lambda$ and the sample mean, respectively.

constant λ ($\hat{p}_n^\lambda$) and the sample mean, respectively. For simplicity, below simply call the SLWE with constant λ for SLWE. The black curve shows the true value of p in each iteration. We see that the test in Theorem 2 detects the changes in p very efficiently such that on average will Algorithm 1 (gray) perform better than SLWE. As expected, the sample mean is not very useful in a dynamic environment.

Figure 3 visualizes a case with smaller changes in p. We see that the changes in p also here will be efficiently detected and that Algorithm 1 perform better than the SLWE. In both experiments above we chose $\alpha = 10^{-3}$, $\lambda = 0.96$ and $\tilde{n} = [1/(1-\lambda)] = 25$.

In Figure 4 we visualize the estimators for an environment where p is changing smoothly. More specifically the true p changes following a cosine function. For such an environment, it seems like $\tilde{p}_n$ and $\hat{p}_n^\lambda$ perform almost equally well. Note that even though the Algorithm 1 is not constructed for such environments, we see that it still performs well. In this experiment we chose $\alpha = 10^{-2}$, $\tilde{n} = 1$ and still $\lambda = 0.96$

In Algorithms 1 and 2 there are three tuning parameters, namely α, $\tilde{n}$ and λ. We now want evaluate how the choices of these parameters affect the tracking performance. For the binomial case we considered the following two different cases:

- Large changes: p changed with time as shown in Figures 2.
- Small changes: p changed with time as shown in Figures 3.

For the multinomial case we considered the two cases:

1. Every $D = 600$ iterations, we changed the probability vector as follows
   - Draw a random number ρ uniformly from $1, 2, \ldots, r$
   - Set $p_\rho = 0.8$
   - Set $p_i = 0.2/(r-1)$ for $i \neq \rho$

   Below we refer to this alternative as 'spike probability'.

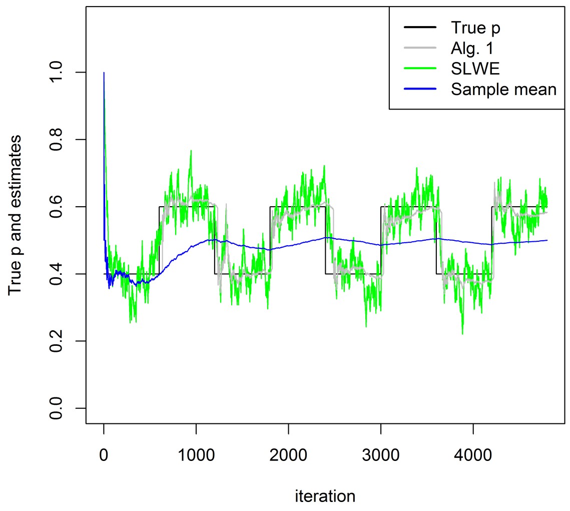

Figure 3: Evaluation of the estimators in an environment with small jumps in p. The black curve shows the true p in each iteration. The gray, green and blue curves show the estimators $\tilde{p}_n$, $\hat{p}_n^\lambda$ and the sample mean, respectively.

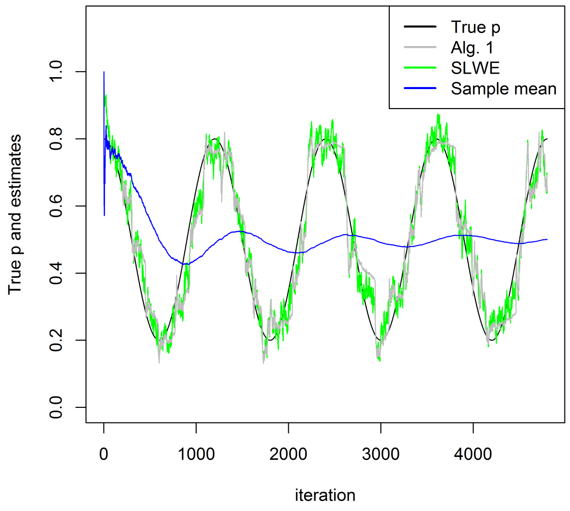

Figure 4: Evaluation of the estimators in an environment where p dynamically is changing. The black curve shows the true p in each iteration. The gray, green and blue curves show the estimators $\tilde{p}_n$, $\hat{p}_n^\lambda$ and the sample mean, respectively.

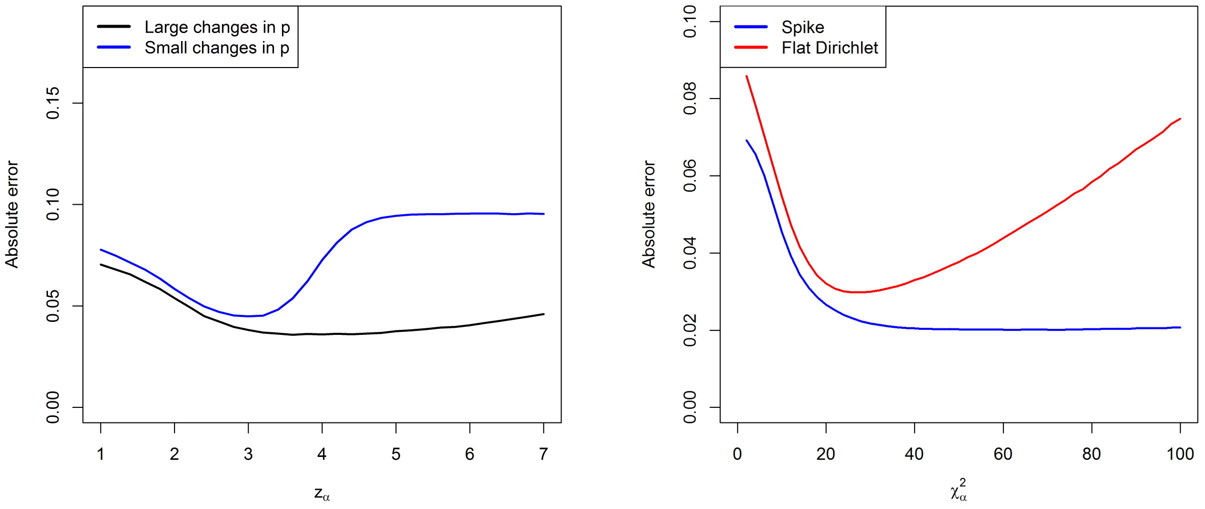

Figure 5: The left and and right panels show estimation error as a function of z_α (binomal case) and $\chi^2_{r-1.\alpha}$ (multinomial case), respectively. For the left panel, the black, blue and red curves refer to experiments where the changes in p are large, small and dynamic. For the right panel, the blue and red curves refer to experiments where the changes in p are based on spike probability and flat Dirichlet alternatives, respectively.

2 Every $D = 600$ iterations, we updated the probability vector as an outcome from the Dirichlet distribution with parameter values $\alpha_1 = 1, \ldots, \alpha_r = 1$. This is referred to as the flat Dirichlet distribution and the probability distribution is uniformly distributed over the simplex of possible probability vectors, i.e. the vectors satisfying $p_1, p_2, \ldots, p_r > 0$ and $\sum_{i=1}^{r} p_i = 1$. Below we refer to this alternative as 'flat Dirichlet'.

We measure the estimation error as the difference in absolute value between the true p and the estimate averaged over all the iterations.

We start by investigating reasonable values for the significance value α. For the binomial and multinomial algorithms we computed the estimation error for different choices of z_α and $\chi^2_{r-1,\alpha}$, respectively. To reduce the Monte Carlo error we ran the Bernoulli and multinomial data streams for $5 \cdot 10^6$ iterations. In the experiments we used $\lambda = 0.95$ and $\tilde{n} = [1/(1-\lambda)] = 20$. We assumed that the system shifted state every $D = 600$ iteration similar to the examples in Figures 2 and 3.

The results are shown in Figure 5. We start discussing the binomial case (left panel). For the blue curve in the left panel of Figure 5 we see that an optimal value of z_α is about 3 which is equivalent to $\alpha \approx 10^{-3}$. By choosing smaller values of z_α, the test will too often wrongly detect changes. Choosing a too high value of z_α, the test will detect changes in p too late or never. With z_α above 5, the test will never detect the changes in p and we reach a limit in the estimation error which is equal to the estimation error using the sample mean. When the changes in p are large (black curve), we can allow using higher values of z_α since we still are able to detect the large changes in p. An optimal value for z_α is around 4. Choosing an even higher value of z_α slightly reduces the performance because the method uses a few iterations more before detecting that p has changed value.

For the multinomial case (right panel), we see that for the flat Dirichlet alternative, an optimal value is $\chi^2_{r-1,\alpha} \approx 25$ which is equivalent to $\alpha \approx 10^{-5}$. For the spike probability alternative any value of $\chi^2_{r-1,\alpha}$ between 25 and 100 perform well. We see that a lower value of α performs the best in the multinomial cases compared to the binomial cases. The reason is that it is easier

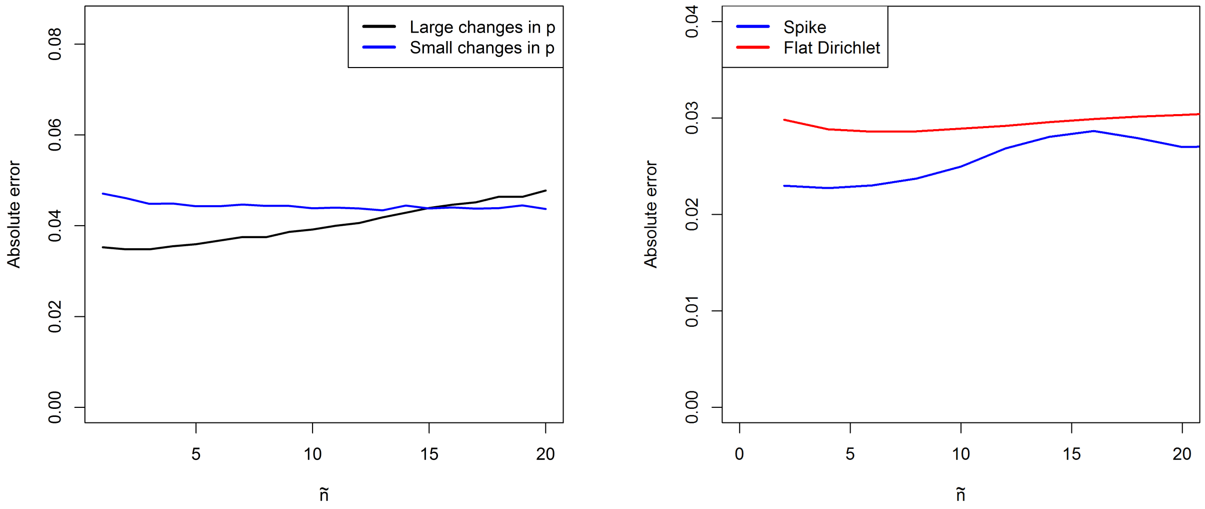

Figure 6: Estimation error as a function of $\tilde{n}$. For the left panel, the black, blue and red curves refer to experiments where the changes in p are large, small and dynamic. For the right panel, the blue and red curves refer to experiments where the changes in p are based on spike probability and flat Dirichlet alternatives, respectively.

to detect a change in the probability vector in the multinomial case compared to the binomial case.

We turn our attention now to evaluating the optimal values for $\tilde{n}$. The results are shown in Figure 6. Also in this experiment we set $\lambda = 0.95$. Further we sat $z_\alpha = 3$ and $\chi^2_{r-1,\alpha} = 25$. Overall we see that the estimation error does not depend strongly on the choice of $\tilde{n}$. Please note that the increase in estimation error for $\tilde{n} \approx 15$ for the spike probability alternative is an actual effect and not Monte Carlo error.

Finally we investigate how the estimation error depends on the choice of λ. We compare the performance of Algorithm 1, the SLWE and the exponentially weighted moving average control scheme in [38], denoted ECDD (EWMA for Concept Drift Detection). ECDD is the algorithm in the literature that is most closely related to our estimator. However, as pointed out in Section 4, the ECDD algorithm has some substantial weaknesses compared to Algorithm 1. To be able to make a proper comparison, we adopt the detection part of the ECDD algorithm into our Algorithm 1, i.e. we change line 9 in Algorithm 1, with the event detection procedure in Table 2 in [38]. Please note that within concept drift as considered in [38], it is natural to only look for increases in p, while in a tracking setting one must look for both increases and decreases. Thus we change the last line in Table 2 in [38] with the absolute value in differences between the estimators as in line 9 in Algorithm 1. The ECDD detection procedure, requires a value for the average run length (ARL0) and we consider ARL0 = 100, 400 and 1000 as in [38].

The results are shown in Figure 7. We start by discussing the large changes in p case (left panel). We see that Algorithm 1 outperforms SLWE with a large margin. We also observe that an optimal value for λ in Algorithm 1 and the SLWE is about 0.9 and 0.96, respectively. This difference may come as a surprise, but remember that the purpose of the weak estimator with constant λ is different for these two cases. For SLWE we chose λ to minimize the estimation error while for the Algorithm 1, we chose λ to detect changes in p as fast as possible to rapidly perform a jump. Further we see that Algorithm 1 outperforms the ECDD algorithm with a clear margin. In the right panel we observe that also for the small changes in p case Algorithm 1 outperforms both the SLWE and the ECDD algorithms.

Figure 8 shows results for the multinomial case. We are not aware of an extension of the

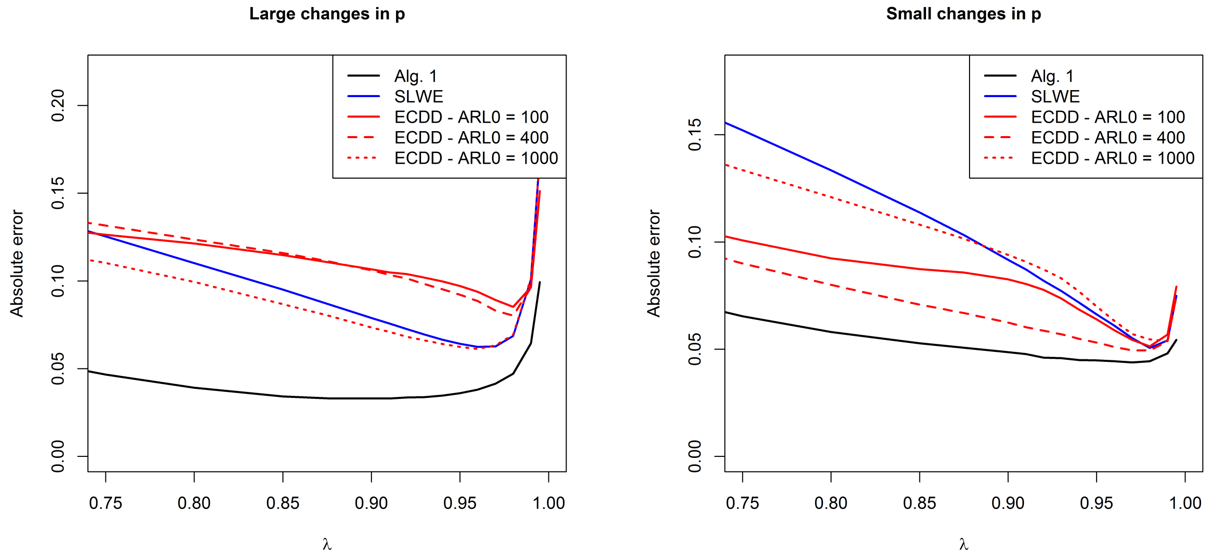

Figure 7: Binomial case: Estimation error as a function of λ for Algorithm 1, the SLWE and ECDD algorithm in [38].

ECDD algorithm to the multinomial case, and we compare Algorithm 2 with the SLWE. We see that Algorithm 2 outperforms the multinomial SLWE with a large margin for both the spike probability and the flat Dirichlet alternatives.

From both Figures 7 and 8, we see that the performance of Algorithms 1 and 2 are less sensitive to the choice of λ compared to the SLWE and the ECDD. In other words, Algorithms 1 and 2 perform well for a large range of different choices of λ while the SLWE and the ECDD are able to perform well only for a small interval of values for λ. This is a very useful property since in real-life settings the optimal value of λ is rarely known in advance.

We now conduct a comparison against several other recently developed algorithms in the literature. We are not aware of many tracking algorithms designed to handle data streams with abrupt changes, but the ADWIN2 approach in [5] and the Fishers Exact Test (FET) approach in [39] are two prominent exceptions. For binomial data streams, the FET approach is appealing from a theoretical point of view since it is based in the exact Fisher test. However, in practice it was very computationally demanding compared to our method. Another and much used approach in abruptly changing data streams, is to combine a tracking procedure with a change detection procedure. When a change is detected, the tracking restarts [17]. We compare against several change detection procedures: the Student t-test statistic in [11], the Mann-Whitney and Lepage test statistics in [37] and the Kolmogorov-Smirnov and Cramer-von-Mises test statistics in [36]. Let m, n and s denote the time of the most recent change detection, the current time point and some time point between m and s. The approaches above run comparisons between the observations in the intervals $[m, s]$ and $[s + 1, n]$ for every $m < s < n$. If any of these tests detects a significant difference, a change is reported. Thus these approaches run several tests in each iteration n, while our approach is far less computationally complex since only one computationally lightweight test needs to be ran in each iteration.

The estimation results are shown in Table 1. For each method, tuning parameters are adjusted to optimize performance. We see that our suggested approach outperforms all the alternative algorithms in the literature. Among the alternative algorithms, the Mann-Whitney and Student t-tests detection procedures performed the best.

We observed that the student t-test performed well in the binomial cases and its natural extension for multivariate data, is the Hotellings T^2 test [15]. A comparison of our multivariate approach in Algorithm 2 against the multinomial SLWE and the Hotellings T^2 change detection

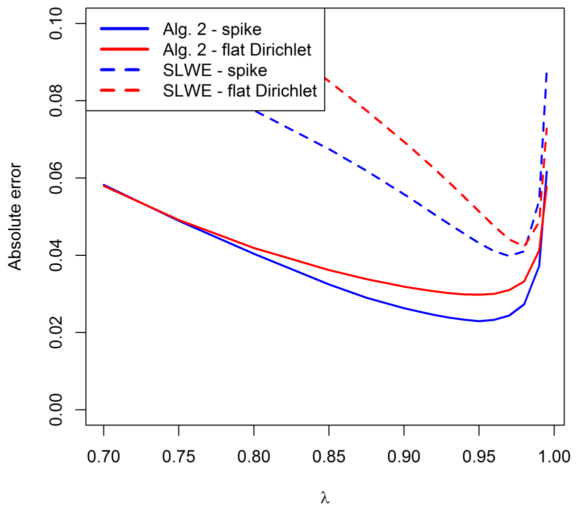

Figure 8: Multinomial case: Estimation error as a function of λ for Algorithm 2 and the multivariate SLWE.

Method	Large changes	Small changes
Algorithm 1	0.033	0.044
SLWE	0.062	0.050
ADWIN2	0.143	0.093
FET	0.076	0.080
ECDD	0.061	0.049
Student	0.037	0.054
Mann-Whitney	0.035	0.053
Lepage	0.133	0.160
Kolmogorov-Smirnov	0.166	0.140
Cramer-von-Mises	0.181	0.117

Table 1: Binomial case: Mean estimation error in absolute value when tracking p for the suggested algorithm and several other algorithms in the literature.

Method	Spike	Flat Dirichlet
Algorithm 2	0.023	0.031
SLWE	0.040	0.041
Hotelling's T^2	0.039	0.038

Table 2: Multinomial case: Mean estimation error in absolute value when tracking p for the suggested algorithm, SLWE and the Hotellings T^2 algorithms in the literature.

approach is shown in Table 2. Similar to the change detection methods above, the Hotelling's T^2 method run a test for each s between m and n and thus is far more computationally expensive than Algorithm 2. We see that Algorithm 2 outperforms both the SLWE and the Hotellings T^2.

6.2 Real-life data example

In this section, we investigate the problem of tracking topics or sentiment in online streams of text. Examples of such text streams could be online discussion threads and news/social media feeds like Twitter. In this specific example we consider the problem of online tracking of the current topic in a news feed. We assumed four topics, namely news about the European Union (EU), economy, sports and entertainment. We collected a large set of news articles about the four topics from the popular Norwegian online news paper site `vg.no`. We assumed that the instants when the text stream changed between the different topics were unknown to our algorithm. The problem was to track the probabilities that the current topic of the stream was EU, economy, sports or entertainment. The problem has been considered in several papers, see e.g. [7, 32] and references therein.

We consider two different approaches:

- **Keyword lists**: A keyword list is a set of words for each topic or sentiment type (for example: happy, sad, angry, etc). We divided the material in two parts, where the first part were used to generate a keyword list for each topic or sentiment. The keyword list for a given topic/sentiment were generated by choosing words that had a high Pointwise mutual information to the given topic/sentiment [26].

  The second part of the material where used for tracking. We assumed that we received one word at the time from the news feed and every time we received a new word, we updated our probability estimates that the current topic were EU, economy, sports or entertainment. If the current word received from the news feed was part of the EU keyword list, we can think of this as an outcome '1' from a multinomial distribution. If the word was part of the economy keyword list, we can think of this as an outcome '2' from a multinomial distribution and so on. Using the weak estimator in equation (9), we can now update our estimate of the probability vector, namely the probabilities that the current topic is EU, economy, sports or entertainment. Similarly we can update the estimate of the probability vector using the jump algorithm in Algorithm 2. If a word where not part of any of the keyword lists, any of the probabilities were changes.

  A natural offline way to estimate of the probability that the current topic was EU (economy, sports, entertainment) based on the keyword lists was to compute the portion of all the keywords in an article that were EU (economy, sports, entertainment) keywords. We denote this the offline approach and can be seen as the optimal estimates for the probability of the different topics based on the keyword lists. In an online setting it is not possible to compute the offline estimates, but ideally we want the online estimators in (9) and in Algorithm 2 to be as close as possible to the optimal offline estimates. We compare the performance of the online estimators in this paper by measuring how close they were to the optimal offline approach. When performing the experiments we ran a two fold cross validation where we used half of the articles to compute the keyword lists and the

Method	Logistic regression	Keyword lists
Algorithm 1	0.035	0.064
SLWE	0.071	0.077
ADWIN2	0.322	0.238
FET	0.109	0.068
ECDD	0.070	0.068
Student	0.043	0.076
Mann-Whitney	0.036	0.062
Lepage	0.119	0.162
Kolmogorov-Smirnov	0.133	0.177
Cramer-von-Mises	0.137	0.181

Table 3: News example, two topics case: Mean estimation error in absolute value compared to offline estimator for the suggested algorithm and several other algorithms in the literature.

Method	Multinomial regression	Keyword lists
Algorithm 2	0.036	0.039
SLWE	0.057	0.056
Hotelling's T^2	0.041	0.065

Table 4: News example, four topics case: Mean estimation error in absolute value compared to offline estimator for the suggested algorithm, SLWE and the Hotellings T^2 algorithms in the literature.

other half to track the probabilities that the current topic was EU, economy, sports or entertainment. Next, we switched and trained and tested in the opposite direction.

- **Machine learning:** We started by dividing the training text stream in batches of 20 words where each batch were within one of the topics EU, economy, sports or entertainment. The batches were used to train a machine learning model and in this example we used multinomial ridge regression [9]. The model were trained using the `glmnet` package in R [33].

  In the testing part, the individual words of the text stream were collected into batches of 20 words. Each new batch were classified into one of the four topics using the trained multinomial regression model. The probabilities of the current topic where updated in the same manner as for the keyword list approach.

Figure 9 shows the tracking of the probabilities for the different topics for the keyword list approach. We see that Algorithm 2 adapts faster when the text stream changes topic and also tracks the offline estimates more efficiently in the stationary parts than the SLWE.

We now perform a more systematic analysis of the performance of Algorithms 1 and 2. We start with a binomial case by merging the topics EU and economy to one class and sport and entertainment to a second class. The approach will be identical to what is described above, except that we only trained two keyword lists and used logistic ridge regression in the machine learning approach. Table 3 shows the tracking performance for the same methods considered in the synthetic experiments. Algorithm 1 documents competitive results to the Mann-Whitney approach and outperforms all the other approaches. Recall that the Mann-Whitney approach is far more computationally intensive than Algorithm 1 to achieve these results.

Next we analyze the performance of different algorithms for the multinomial case with all the four topics. The results are summarized in Table 4. We see that Algorithm 2 outperforms both the SLWE and the Hotelling's T^2 algorithms.

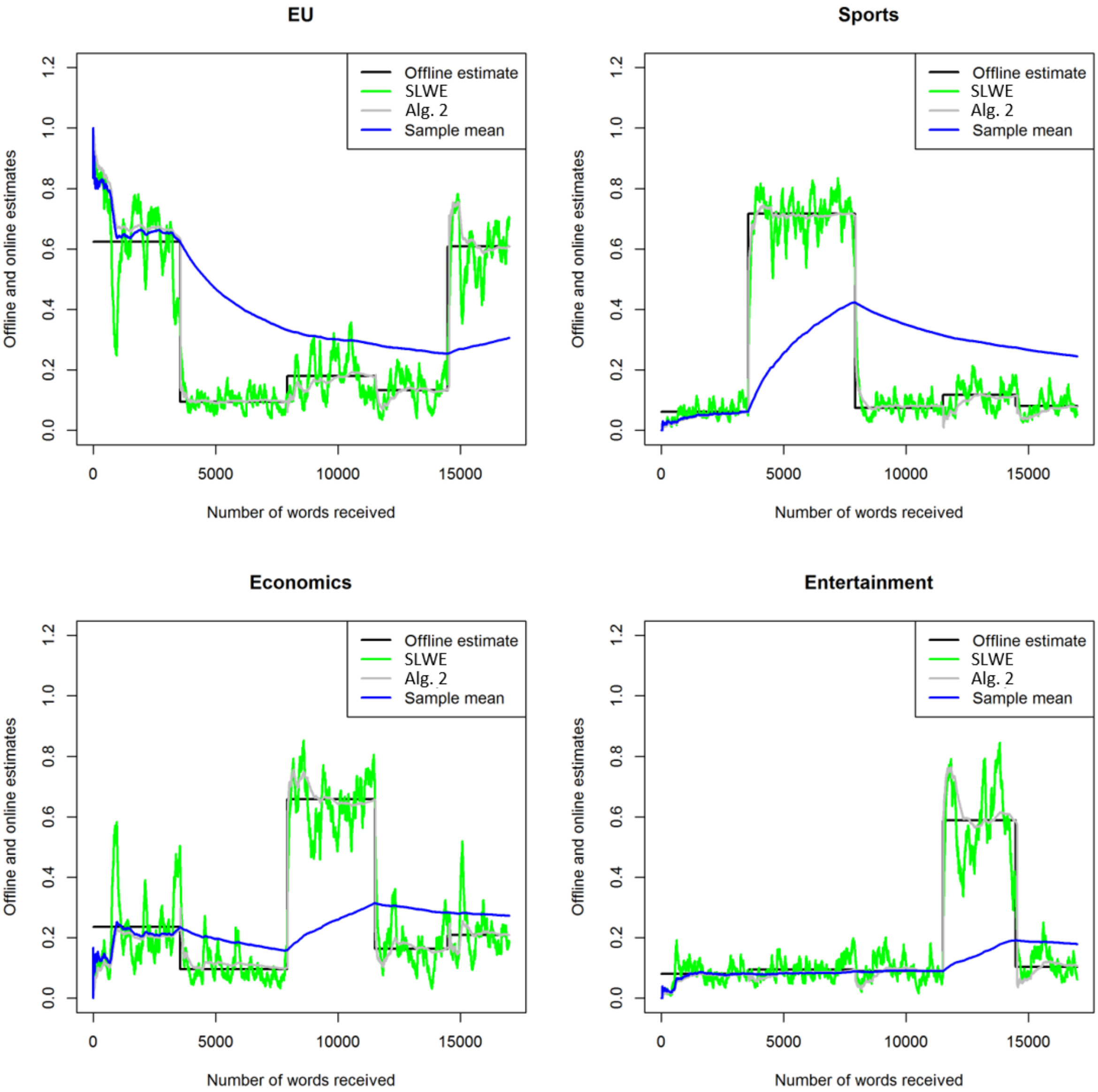

Figure 9: The panels from upper left to bottom right show the tracking of the probabilities that the current topic is EU, economy, sports or entertainment, respectively. The black curves shows the offline estimate every time a new word is received. The gray, green and blue curves show the Algorithm 1, the SLWE estimator and the sample mean, respectively.

7 Closing remarks

In this paper we have constructed an estimation procedure that combines the strengths of a weak estimator with constant λ and and decreasing λ (sample mean). We have developed a hypothesis test procedure to rapidly detect a change in the underlying p. Further we have proposed an efficient procedure to jump to a new estimate when a change is detected. The experiments show that the procedure efficiently detects changes in the underlying distribution and outperforms the original SLWE and several other algorithms from the literature.

The experiments also showed that the performance of the Algorithm 1 $\tilde{p}_n$ is less sensitive to the choice of λ compared to the SLWE with constant λ and the ECDD algorithm. Said in another way, the Algorithm 1 performs well for a large range of different choices of λ while the SLWE and ECDD performed well only for a small range of choices for λ. This is a very attractive property of the Algorithm 1 since in practical situations we do not know what is an optimal value for λ.

In the χ^2 testing procedure in Theorem 3, the dependency between dimensions is ignored. A potential direction for future research is to develop and analyze a test where the dependency is taken into account. The covariances computed for the multivariate exponentially weighted control scheme may be a good starting point [24].

A potential direction for future research is to develop a procedure to automatically adjust the value of the tuning parameter λ depending on the properties of the data stream. A starting point in this direction could be to follow the ideas on how the ADWIN2 algorithm adjusts the window size [5].

References

[1] Alan Agresti and Maria Kateri. *Categorical data analysis*. Springer, 2011.

[2] Michle Basseville and Igor V. Nikiforov. *Detection of abrupt changes: theory and application*. Prentice-Hall, Inc., 1993.

[3] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 289–300, 1995.

[4] Peter J Bickel and Kjell A Doksum. *Mathematical statistics: basic ideas and selected topics*. CRC Press, 2015.

[5] Albert Bifet and Ricard Gavalda. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 443–448. SIAM, 2007.

[6] Tamraparni Dasu, Shankar Krishnan, Suresh Venkatasubramanian, and Ke Yi. An information-theoretic approach to detecting changes in multi-dimensional data streams. In *In Proc. Symp. on the Interface of Statistics, Computing Science, and Applications*. Citeseer, 2006.

[7] Massimo De Santo, Gennaro Percannella, Carlo Sansone, and Mario Vento. A multi-expert approach for shot classification in news videos. *Image Analysis and Recognition*, pages 564–571, 2004.

[8] Anton Dries and Ulrich Rückert. Adaptive concept drift detection. *Stat. Anal. Data Min.*, 2(5):311–327, December 2009.

[9] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

[10] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4):44:1–44:37, March 2014.

[11] Douglas M Hawkins, Peihua Qiu, and Chang Wook Kang. The changepoint model for statistical process control. *Journal of quality technology*, 35(4):355, 2003.

[12] Douglas M Hawkins and KD Zamba. A change-point model for a shift in variance. *Journal of Quality Technology*, 37(1):21, 2005.

[13] Douglas M Hawkins and KD Zamba. Statistical process control for shifts in mean or variance using a changepoint formulation. *Technometrics*, 47(2):164–173, 2005.

[14] Amin Ibrahim and Miguel Vargas Martin. Detecting and preventing the electronic transmission of illicit images and its network performance. In *International Conference on Digital Forensics and Cyber Crime*, pages 139–150. Springer, 2009.

[15] Richard Arnold Johnson, Dean W Wichern, et al. *Applied multivariate statistical analysis*, volume 4. Prentice-Hall New Jersey, 2014.

[16] A. F. Karr. *Probability.* Springer, New York, 2012.

[17] Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 180–191. VLDB Endowment, 2004.

[18] Ralf Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intell. Data Anal.*, 8(3):281–300, August 2004.

[19] Ivan Koychev. Gradual forgetting for adaptation to concept drift. In *Proceedings of ECAI 2000 Workshop Current Issues in Spatio-Temporal Reasoning*, pages 101–106, 2000.

[20] Ivan Koychev. Gradual forgetting for adaptation to concept drift. Proceedings of ECAI 2000 Workshop on Current Issues in Spatio-Temporal Reasoning,, 2000.

[21] Ivan Koychev and Robert Lothian. Tracking drifting concepts by time window optimisation. In Max Bramer, Frans Coenen, and Tony Allen, editors, *Research and Development in Intelligent Systems XXII*, pages 46–59. Springer London, 2006.

[22] Ivan Koychev and Ingo Schwab. Adaptation to drifting user's interests. In *Proceedings of ECML2000 Workshop: Machine Learning in New Information Age*, pages 39–46, 2000.

[23] Pallavi Kulkarni and Roshani Ade. Incremental learning from unbalanced data with concept class, concept drift and missing features: A review. *International Journal of Data Mining and Knowledge Management Process (IJDKP)*, 4(6):15–29, November 2014.

[24] Cynthia A Lowry, William H Woodall, Charles W Champ, and Steven E Rigdon. A multivariate exponentially weighted moving average control chart. *Technometrics*, 34(1):46–53, 1992.

[25] James M Lucas and Michael S Saccucci. Exponentially weighted moving average control schemes: properties and enhancements. *Technometrics*, 32(1):1–12, 1990.

[26] Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*, volume 999. MIT Press, 1999.

[27] Sudip Misra, Nayan Ranjan Kapri, and Bernd E Wolfinger. Selfishness-aware target tracking in vehicular mobile wimax networks. *Telecommunication Systems*, 58(4):313–328, 2015.

[28] Ratish Mohan, Anis Yazidi, Boning Feng, and B John Oommen. Dynamic ordering of firewall rules using a novel swapping window-based paradigm. In *Proceedings of the 6th International Conference on Communication and Network Security*, pages 11–20. ACM, 2016.

[29] Kumpati S Narendra and Mandayam AL Thathachar. *Learning automata: an introduction.* Courier Corporation, 2012.

[30] B. J. Oommen and S. Misra. Fault-tolerant routing in adversarial mobile ad hoc networks: an efficient route estimation scheme for non-stationary environments. *Telecommunication Systems*, 44:159–169, 2010.

[31] B. J. Oommen, A. Yazidi, and O-C. Granmo. An adaptive approach to learning the preferences of users in a social network using weak estimators. *Journal of Information Processing Systems*, 8(2), 2012.

[32] B. John Oommen and Luis Rueda. Stochastic learning-based weak estimation of multinomial random variables and its applications to pattern recognition in non-stationary environments. *Pattern Recogn.*, 39(3):328–341, 2006.

[33] R Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2017.

[34] Nasser-Eddine Rikli and Aljawharah Alnasser. Lightweight trust model for the detection of concealed malicious nodes in sparse wireless ad hoc networks. *International Journal of Distributed Sensor Networks*, 12(7):1550147716657246, 2016.

[35] Gordon J Ross. Sequential change detection in the presence of unknown parameters. *Statistics and Computing*, 24(6):1017–1030, 2014.

[36] Gordon J Ross and Niall M Adams. Two nonparametric control charts for detecting arbitrary distribution changes. *Journal of Quality Technology*, 44(2):102, 2012.

[37] Gordon J Ross, Niall M Adams, Dimitris K Tasoulis, and David J Hand. A nonparametric change point model for streaming data. *Technometrics*, 53(4):379–389, 2011.

[38] Gordon J. Ross, Niall M. Adams, Dimitris K. Tasoulis, and David J. Hand. Exponentially weighted moving average charts for detecting concept drift. *Pattern Recognition Letters*, 33(2):191 – 198, 2012.

[39] Gordon J Ross, Dimitris K Tasoulis, and Niall M Adams. Sequential monitoring of a bernoulli sequence when the pre-change parameter is unknown. *Computational Statistics*, pages 1–17, 2013.

[40] L. Rueda and B. John Oommen. Stochastic automata-based estimators for adaptively compressing files with nonstationary distributions. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 36(5):1196 –1200, October 2006.

[41] Raquel Sebastião and João Gama. Change detection in learning histograms from data streams. In *Proceedings of the Aritficial Intelligence 13th Portuguese Conference on Progress in Artificial Intelligence*, EPIA'07, pages 112–123, Berlin, Heidelberg, 2007. Springer-Verlag.

[42] Albert Nikolaevich Shiryayev. *Optimal Stopping Rules.* Springer, 1978.

[43] A. Stensby, B. J. Oommen, and O-C. Granmo. The use of weak estimators to achieve language detection and tracking in multilingual documents. *International Journal of Pattern Recognition and Artificial Intelligence*, 27(04):1350011, 2013.

[44] A. G. Tartakovsky, B. L. Rozovskii, R. B. Blazek, and Hongjoong Kim. 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*, 54:3372–3382, September 2006.

[45] Alexey Tsymbal, Mykola Pechenizkiy, Pádraig Cunningham, and Seppo Puuronen. Dynamic integration of classifiers for handling concept drift. *Inf. Fusion*, 9(1):56–68, January 2008.

[46] Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

[47] A. Yazidi, O-C. Granmo, B. J. Oommen, M. Gerdes, and F. Reichert. A user-centric approach for personalized service provisioning in pervasive environments. *Wireless Personal Communications*, 61(3):543–566, 2011.

[48] Anis Yazidi and B. John Oommen. Novel discretized weak estimators based on the principles of the stochastic search on the line problem. *IEEE Trans. Cybernetics*, 46(12):2732–2744, 2016.

[49] Anis Yazidi, B John Oommen, Geir Horn, and Ole-Christoffer Granmo. Stochastic discretized learning-based weak estimation: a novel estimation method for non-stationary environments. *Pattern Recognition*, 60:430–443, 2016.

[50] Justin Zhan, B. John Oommen, and Johanna Crisostomo. Anomaly detection in dynamic systems using weak estimators. *ACM Trans. Internet Technol.*, 11:3:1–3:16, July 2011.