Estimation, forecasting and anomaly detection for nonstationary streams using adaptive estimation

Henrique Hoeltgebaum, Niall Adams, Cristiano Fernandes

***Abstract*—Streaming data provides substantial challenges for data analysis. From a computational standpoint, these challenges arise from constraints related to computer memory and processing speed. Statistically, the challenges relate to constructing procedures that can handle so-called *concept drift* – the tendency of future data to have different underlying properties to current and historic data. The issue of handling structure, such as trend and periodicity, remains a difficult problem for streaming estimation. We propose *RAC* (Real-Time Adaptive Component), a penalized-regression modelling framework which satisfies the computational constraints of streaming data, and provides capability for dealing with concept drift. At the core of the estimation process are techniques from adaptive filtering. The RAC procedure adopts a specified basis to handle local structure, along with a LASSO-like penalty procedure to handle over-fitting. We enhance the RAC estimation procedure with a streaming anomaly detection capability. Experiments with simulated data suggest the procedure can be considered as a competitive tool for a variety of scenarios, and an illustration with real cyber-security data further demonstrates the promise of the method.**

***Index Terms*—Adaptive filtering, data stream, time-varying sparsity, anomaly detection, forgetting factor.**

I. Introduction

STREAMING data – an unending sequence of data values arriving at high frequency – is becoming ubiquitous due to advances in data acquisition technology [1], [2]. There is a clear demand for the development of streaming statistical methods, considering applications in diverse areas such as cyber-security [3], predictive maintenance systems [4], [5], finance [6], fraud detection [7] and structural health monitoring [8]. Such data brings significant challenges, related to both demands arising from sequential computation and the design of suitable estimators [9]–[11].

An outstanding, open, problem relates to handling structure, such as trend or seasonality, in the data stream. For the batch case, there is a large arsenal of time series and related tools available to address such issues, e.g., the Kalman Filter [12]. However, in the streaming case the data is revealed sequentially, with the risk that statistical properties of the data may vary over time. This is known as concept drift [1], [13], which invalidates the use of methods that assume various modes of stationarity.

For example, [14] considered the objective of characterizing and forecasting an *arbitrary* streaming data sequence. These authors made use of a partially observed Markov process, where the evolution of the latent state is governed by a continuous-time Markov process, which allows modelling of irregularly spaced observations. The justification for irregular spacing arises from the sampling frequency of sensors which are constantly interrupted and re-started. To capture the dynamics of all possible latent state variables that constitute the underlying structure of the stream, [14, Sec. 3] described a very elegant way to combine models. As an example, they used a composition of a negative binomial model with two seasonal components, considering daily and weekly seasonality effects respectively, to fit vehicle traffic monitoring data. However, a potential shortcoming of their approach, is that the analyst must use prior knowledge, for instance to specify seasonal components (daily and weekly). This means that any behaviour outside such specification cannot be accommodated. This potentially limits the application of the method for tracking an infinite data stream subject to random fluctuations and concept drift.

One approach which can, in principle, cope with arbitrary structure in the data stream is based on estimation with Adaptive Forgetting Factors (AFF) e.g. [15]–[19]. The use of stochastic gradient descent [16] to update a Forgetting Factor (FF) enables models to handle arbitrary changes in the data generating process. The FF is intended to down weight historic data in the estimation process. To cope with concept drift, we will utilise adaptive estimation methods in a number of ways, with the intention of constructing a streaming regression model for a univariate response, where the explanatory variables can be regarded as local auto-regressive terms. This model is well suited to the streaming context, in terms of data storage and computational requirements, and offers a choice of basis for the auto-regressive regression.

The proposed method, which we refer to as Real-time Adaptive Component (RAC), relies on a penalised streaming regression-based framework. A simple form of the streaming setting, described in [20], for a linear regression model is

$$y_t = x_t^{'}\beta_t + \varepsilon_t,$$

where $\varepsilon_t \sim N(0, \sigma^2)$ are independent and identically distributed (*i.i.d.*) Gaussian random variables with known variance σ^2. Data processing typically proceeds as follows: Consider the tick t, acquire basis vector $x_t \in \mathbb{R}^d$ and use it to forecast $\hat{y}_t \in \mathbb{R}$, the one step ahead forecast, using the sequentially estimated weight vector $\hat{\beta}_t \in \mathbb{R}^d$. Later, with the acquisition of the true value y_t, $\hat{\beta}_t$ is updated to $\hat{\beta}_{t+1}$. In our case, the basis vector represents lagged and transformed values of the response.

The work of H. Hoeltgebaum was fully supported by the National Council for Research and Development, CNPq, Ministry of Science and Technology, Brazil. H. Hoeltgebaum and Niall Adams are with the Mathematics Department and Data Science Institute – Imperial College London (hh3015@ic.ac.uk, n.adams@imperial.ac.uk). Cristiano Fernandes is with the Electrical Engineering Department, Pontifical Catholic University of Rio de Janeiro, Brazil (cris@ele.puc-rio.br). *(Corresponding author: H. Hoeltgebaum).*

In the batch case, extracting information from time series is receiving much recent attention [21]–[24]. In the streaming context however, the choice of basis for the regression, or equivalently, the set of transformations and lagged variables is critical for successful forecasting, and hence anomaly detection. However, since streaming data is subject to unpredictable temporal variation, the construction of a suitable basis is challenging. There is the usual problem of ensuring the model has sufficient complexity to capture underlying structure, while not affording the opportunity for over-fitting [20], [25].

This work proposes the use of a relatively large basis (the maximum dimension defined by computational constraints), and then appeals to sparsity inducing approaches to manage over-fitting. We use a scheme based on the Least Absolute Shrinkage Operator (LASSO) technique, proposed by [26] and further detailed in [27]. A vast literature showing that the LASSO attains good performance under various assumptions on the basis is available (see [28], [29] and references therein). In our context, the appealing aspect of the LASSO is that it induces sparsity, which will provide a means for managing over-fitting. To achieve this, we require a method for sequentially determining the LASSO penalisation parameter. This issue was addressed in [30], which proposed an adaptive extension of a LASSO Vector Autoregressive (VAR) model to perform hourly wind power forecasts for wind farms. Similar to AFF, the authors also used FF to handle the nonstationary of the signal, albeit in their version it is a fixed quantity. Unfortunately, despite the autoregressive coefficients of the VAR being updated sequentially, the penalty term is not. By default, these models are fitted using a grid of penalty parameters which are computationally unfeasible in streaming data context. Similarly, [31] proposed an updating rule to the penalty term designed to estimate the parameters as soon as new data arrives and assuming the underlying distribution is nonstationary. Similar to AFF, the framework of [31] also adopts adaptive filtering estimation based on stochastic gradient descent.

The need for conditional models that can cope with concept drift is well recognised by the streaming data community (see [9], [32], [33] and references therein). The contribution of this paper is the development of a forecasting procedure and associated *local* anomaly detector, capable of dealing with the many challenges of streaming data. This is achieved by extending the results of [31]. First, proposing two new adaptive estimators for a penalized-regression model. Second, adopting a basis construction strategy to track the evolution of signals' underlying structure. Third, developing a new method to sequentially perform anomaly detection in streaming data that can be used with any conditional model, using a streaming method based on the results of [34].

The rest of this paper is organised as follows. In Section II, the main features of RAC along with the proposed anomaly detection procedure are introduced. In Section III, a simulation study is provided to evaluate both estimation and detection performance of the RAC method. In Section IV a case study using real cyber-security data from Los Alamos National Laboratory is conducted.

II. Methodology

In this section the basic components of RAC are introduced. Specifically the LASSO, adaptive estimation, optimization procedures, basis construction and extension to anomaly detection are discussed.

A. The LASSO procedure

The batch LASSO estimator [26] was initially proposed as a variable selection procedure. Considering the pair (X, y), where y denotes a T-dimensional response vector and X be a $T \times d$ basis, with rows composed by the vectors $x_t \in \mathbb{R}^d$, define the simple linear regression model

$$y = X^{'}\beta + \varepsilon, \tag{1}$$

with weight vector $\beta \in \mathbb{R}^d$ and ε being *i.i.d.* Gaussian random variables with known variance σ^2. Then the LASSO estimator is defined as

$$\hat{\beta}(\gamma) = \arg\min_{\beta} \sum_{t=1}^{T} (y_t - x_t^{'}\beta)^2 + \gamma ||\beta||_1, \tag{2}$$

where $\gamma \geq 0$ is the penalty parameter and $||\cdot||_1$ denotes the ℓ_1 norm. Note that we write $\hat{\beta}(\gamma)$ to emphasise the dependence on γ in the estimation of β. Denote the set of variables as $\mathcal{J} = \{1, ..., d\}$, define the active set of variables, i.e., which variables are selected, as $\mathcal{A}(\gamma) = \{j \in \mathcal{J} : \hat{\beta}^{(j)}(\gamma) \neq 0\}$, where $\hat{\beta}^{(j)}(\gamma)$ makes reference to the jth element of the vector. In practice the solutions $\hat{\beta}(\gamma)$ are estimated on a grid of γ values, ranging from 0, where no shrinkage is applied, to

$$\gamma^{(max)} = \max_{j \in \mathcal{J}} \left| \frac{1}{T} X^{(\cdot,j)^{'}} y \right|, \tag{3}$$

where $X^{(\cdot,j)}$ denotes the jth column of X, and for which all values of $\hat{\beta}(\gamma)$ will be exactly zero, except the intercept. Selection of the penalty parameter is often made through data reuse methods, for example cross-validation (CV), however this is not feasible for streaming data analysis due to computational speed requirements.

Typically, prior to estimation, it is convenient to standardise the data such that each transformed basis vector has zero mean and unit variance. This is done to prevent the LASSO solution from depending on the predictor's units of measurement. In addition, the response variable is centred to have zero mean. These centering operations allows one to omit the intercept term $\beta^{(0)}$ when optimizing (2). Given the optimal LASSO solution $\hat{\beta}(\gamma)$ on the centred data, it is possible to recover the optimal solutions for the uncentred data, where $\hat{\beta}(\gamma)$ remains the same and the intercept is

$$\hat{\beta}^{(0)} = \bar{y} - \sum_{j=1}^{d} \bar{X}^{(j)} \hat{\beta}^{(j)}(\gamma), \tag{4}$$

where $\bar{y}$ is the mean of the untransformed response variable and $\bar{X}^{(j)}$ denotes the mean of the jth transformed basis variables. In the context of streaming data, computing these values are challenging. Despite having useful properties, the batch LASSO estimator is not feasible in streaming data environment. In the next section we introduce adaptive estimation

[16], which can be used to adapt the results of batch LASSO to a streaming data environment, respecting memory and speed constraints.

B. Adaptive estimation

The task of filtering corresponds to controlling the rate at which past information is discarded while avoiding storing all the data in memory. The most common filtering strategy discards information at a constant rate, fixing the value of the FF – denoted here by λ. Adaptive filtering methods do not require the data to be stationary [16]. As an alternative to a fixed value of λ, which may be difficult to set, much interest has focused on sequentially selecting an AFF – λ_t, using an updating mechanism based on stochastic gradient descent [15]–[17]. Such methods are called adaptive because the quantity of data discarded is not constant over time. Particularly, the benefits of using such a strategy are highly relevant in nonstationary environment.

With a univariate stream, $y_1, y_2, ..., y_{T-1}, y_T, ...,$, our goal is to accurately estimate the mean, $E[y_T]$, at tick T. An estimator will be used to detect anomalous behaviour in the stream. For example, the arithmetic mean,

$$\bar{y}_T = \frac{1}{T}\sum_{t=1}^{T} y_t. \tag{5}$$

This estimator makes sense only if $E[Y_t] = \mu$, a constant for all time points. However, denoting τ^* as the change point instant, if there was a change at $\tau^* < T$ such that

$$E[Y_t] = \begin{cases} \mu^{'},\ t = 1, 2, ..., \tau^* \\ \mu,\ t = \tau^* + 1, \tau^* + 2, ..., T, ... \end{cases} \quad , \mu^{'} \neq \mu$$

the arithmetic mean $\hat{\mu} = \bar{y}_T$ cannot estimate μ accurately if there is a big difference between $\mu^{'}$ and μ. In order to improve the estimation, one could take the mean of those observations that occur only after the change point τ^*,

$$\hat{\mu} = \frac{1}{T - \tau^*}\left[y_{\tau^*+1} + y_{\tau^*+2} + ... + y_{\tau^*+T}\right].$$

However, the point τ^* is unknown, which makes this estimation unfeasible in sequential settings.

Such drawback, motivates the use of adaptive estimation to calculate the current mean process at tick t, in which more weight is placed on more recent observations, and do not store all data in memory. Using such methods results in better/improved estimation of the data stream after the change point τ^* [15]. This is achieved by introducing an FF, $\lambda \in (0, 1)$, in (5) and using a normalizing constant $(w_{t,\lambda})$ to weight the estimation process,

$$\bar{y}_{t,\lambda} = \frac{1}{w_{t,\lambda}}\sum_{i=1}^{t} \lambda^{t-i} y_i, \qquad w_{t,\lambda} = \sum_{i=1}^{t} \lambda^{t-i}.$$

The advantage of this formulation is that it leads to a sequential formulation for streaming contexts by defining the following updating mechanism for $t \geq 1$,

$$m_{t,\lambda} = \lambda m_{t-1,\lambda} + y_t \tag{6}$$

$$w_{t,\lambda} = \lambda w_{t-1,\lambda} + 1 \tag{7}$$

$$\bar{y}_{t,\lambda} = \frac{m_{t,\lambda}}{w_{t,\lambda}}, \tag{8}$$

with $m_{0,\lambda} = w_{0,\lambda} = 0$. Note that setting $\lambda = 0$ corresponds to forgetting all previous observations, and only using the most recent observation, i.e. $\bar{y}_{t,\lambda} = y_t$. On the other hand, $\lambda = 1$ corresponds to no forgetting, and then the FF mean, $\bar{y}_{t,\lambda}$, is simply the usual arithmetic mean given in (5). Note that practical algorithms restrict the range of λ to prevent it becoming too small, see for example [15]. The updating mechanism of (6)-(8) is asymptotically related to the Exponential Weighted Moving Average equations, as proved in [15].

The previous updating mechanism extends readily to the linear regression framework of (1). The streaming framework assumes y_t arrives immediately following x_t and preceeding x_{t+1} ([1], [2]). This framework is only relevant to application domains satisfying this data arrival order, such as finance [6] and energy [17]. Other domains exhibit data with more complicated, or random arrival order, leading to the *delayed labelling* problem [35]. To achieve this, we require FF estimates of both mean and covariance of response y_t and basis vector x_t at each tick t. In [15], [17], [31], an adaptive estimation framework was used for both sample mean vector, and sample covariance matrix. Define $\Pi_t = \left(y_t, x_t^{(1)}, x_t^{(2)}, ..., x_t^{(d)}\right)^{'} \in \mathbb{R}^{d+1}$ as the sample mean vector and $\Sigma_{t,\lambda} \in \mathbb{R}^{(d+1)\times(d+1)}$ the sample covariance matrix. Considering a fixed FF λ, the mean vector is updated as

$$\bar{\Pi}_{t,\lambda} = \left(1 - \frac{1}{w_{t,\lambda}}\right)\bar{\Pi}_{t-1,\lambda} + \frac{1}{w_{t,\lambda}}\Pi_t, \tag{9}$$

with $w_{t,\lambda}$ a normalizing constant defined in (7) and $\Pi_{0,\lambda} = (0, 0, ..., 0)^{'}$ a vector of zeros. Note that this is equivalent to applying recursions (6)-(8) to y_t and each element of the d-dimensional vector x_t individually. Further, the covariance matrix $\Sigma_{t,\lambda}$ is updated as

$$\Sigma_{t,\lambda} = \left(1 - \frac{1}{w_{t,\lambda}}\right)\Sigma_{t-1,\lambda} + \frac{1}{w_{t,\lambda}}(\Pi_t - \bar{\Pi}_{t,\lambda})^{'}(\Pi_t - \bar{\Pi}_{t,\lambda}), \tag{10}$$

taking $\Sigma_{0,\lambda}$ as the identity matrix. The effect of these initial values will vanish when adopting a *burn-in* period of $\mathcal{B}$ observations. Similar to the sequential algorithms proposed in [15], [17], [31], assuming that observations $y_1, ...y_{\mathcal{B}}$ will not face any change in the underlying structure, this set of observations will be used to estimate the initial values.

C. Streaming LASSO

RAC relies on a penalised streaming regression-based framework. However, as noted earlier, should the stream manifest concept drift then weighted estimation procedures, described in Section II-B, are clearly appropriate.

The formulation of linear regression in this context would feature coefficients analogous to autoregressive weights in a time series model, called *tap* weights in adaptive filtering [16]. Restricting to linear forms for the regression is restrictive and cannot readily handle trend or seasonality. Thus, we will design bases for streaming regression with the potential to capture these phenomena. These bases are potentially overparametrized and hence streaming penalization methods, adapted from procedures such as LASSO, are required. In this case, the choice of an optimal regularization parameter may itself be time-varying [31]. In order to sequentially fit the underlying structure of y_t, a time-varying penalty parameter $\gamma_t \in \mathbb{R}^+$ is introduced (c.f. (2)).

The regularization parameter is iteratively updated as

$$\gamma_{t+1} = \gamma_t - \eta_\gamma \frac{\partial \mathcal{C}_{t+1}}{\partial \gamma_t}, \tag{11}$$

where $\mathcal{C}_{t+1} = ||y_{t+1} - x_{t+1}^{'}\hat{\beta}_t(\gamma_t)||_2^2$ is the designated cost function to update the stochastic gradient associated with γ_t, while $\eta_\gamma > 0$ is the step size. The intuition behind this update is that the gradient will point the regularization parameter γ_t to move in a direction to minimize the prediction error. If the error increases, the parameter will increase such that the weights $\hat{\beta}_t(\gamma_t)$ will adapt to the new underlying structure of the signal. Here – also in [31] – a quadratic cost function is adopted since the future mean behaviour of y_t is currently being tracked. Nevertheless, other cost functions could also be used, conditionally to the quantity being tracked. For example, [17] define efficient update equations, based on maximum likelihood, for the exponential family of distributions.

To calculate the derivative $\frac{\partial \mathcal{C}_{t+1}}{\partial \gamma_t}$ from (11) using implicit differentiation,

$$\frac{\partial \mathcal{C}_{t+1}}{\partial \gamma_t} = \frac{\partial \mathcal{C}_{t+1}}{\partial \hat{\beta}_t(\gamma_t)} \frac{\partial \hat{\beta}_t(\gamma_t)}{\partial \gamma_t}. \tag{12}$$

While the first term of the right hand side of (12) is straightforward to obtain by direct differentiation, the second is more involving. Under a squared error loss function and ℓ_1-norm of β, [36] showed that for the LASSO, the optimal coefficient path is piecewise linear, which implies that $\partial\hat{\beta}_t(\gamma_t)/\partial\gamma_t$ is piecewise constant. A closed-form solution for this derivative, adapted from [37], is presented in [31, Proposition 1] as

$$\frac{\partial \hat{\beta}_t(\gamma_t)}{\partial \gamma_t} = -(x_t^{'}x_t)^{-1}\text{sign}(\hat{\beta}_t(\gamma_t)) \tag{13}$$

$$= -(\Sigma_{t,\lambda})^{-1}\text{sign}(\hat{\beta}_t(\gamma_t)). \tag{14}$$

This result is equivalent to calculating the gradient of the LASSO solution as suggested by the Least Angle Regression (LARS) formulae in [36, Eqs. (2.4)-(2.6)].

In addition, one should note that similar to the LARS gradient update, (14) is only nonzero over the active set $\mathcal{A}_t(\gamma_t) = \{j \in \mathcal{J} : \hat{\beta}_t^{(j)}(\gamma_t) \neq 0\}$ of regression weights, and zero elsewhere. Note that the subscript t was added to emphasize that this is a time-varying active set. Therefore, at each update of $\partial\hat{\beta}_t(\gamma_t)/\partial\gamma_t$, one should consider the two scenarios described in [31] regarding non-empty active set, $\mathcal{A}_t(\gamma_t) \neq \emptyset$ and empty one, $\mathcal{A}_t(\gamma_t) = \emptyset$):

- Non-empty active set, $\mathcal{A}_t(\gamma_t) \neq \emptyset$, which in this case, as proved in [31], equation (14) is well-defined;
- The active set is empty, $\mathcal{A}_t(\gamma_t) = \emptyset$, then both algorithms, LARS and the one of [31], take a step in the direction of the most correlated predictor $\hat{j} = \underset{j \in \mathcal{J}}{\arg\max} \left\{ |\sum_{t=1}^{T} y_t x_t^{(j)}| \right\}$. Hence define the gradient as

$$\left(\frac{\partial \hat{\beta}_t(\gamma_t)}{\partial \gamma_t} \right)^{(l)} = \delta_{\hat{j}}^{(l)} \text{sign} \left(\sum_{t=1}^{T} y_t x_t^{(l)} \right),$$

  where δ is the Kronecker delta function. All entries of $\partial\hat{\beta}_t(\gamma_t)/\partial\gamma_t$ will be zero with exception of the corresponding to the most correlated predictor.

Note that one could also include an AFF for the parameter estimates, Π_t and Σ_t, which can be concurrently updated with γ_t just by calculating

$$\lambda_{t+1} = \lambda_t - \eta_\lambda \frac{\partial \mathcal{L}_{t+1}}{\partial \lambda_t},$$

where $\mathcal{L}_{t+1}$ can be the squared loss as assumed in [15]. However, non reported experiments suggests that the forecasting performance is dramatically degraded when compared to the fixed FF approach. This is related to the fact that λ_t and γ_t interact, as described in [30]. The penalization parameter may increase to adapt the regression to, for example, a new regime while the AFF value will be reduced to give weight to only recent observations. Both parameters updates are attempting to adapt the model to the new regime. Therefore, we opt to keep λ fixed and update only γ_t as a time-varying quantity in RAC framework.

Adopting the concepts of adaptive filtering discussed in Section II-B, both (9) and (10) are suitable for streaming data as they require storing only a few parameters and data points in computer memory, instead of all historical values. These concepts of adaptive filtering provide grounds to propose sequential updates for (3) and (4). The maximum value of the penalty can be defined as

$$\gamma_t^{(\max)} = \max \left\{ \left| \bar{\Pi}_{t,\lambda}^{(1)} \bar{\Pi}_{t,\lambda}^{(2)} \right|, \left| \bar{\Pi}_{t,\lambda}^{(1)} \bar{\Pi}_{t,\lambda}^{(3)} \right|, ..., \left| \bar{\Pi}_{t,\lambda}^{(1)} \bar{\Pi}_{t,\lambda}^{(d+1)} \right| \right\}, \tag{15}$$

with $|\cdot|$ denoting the absolute value. Note that equation (15) rescales the weighted means by the response variable, i.e., $\bar{\Pi}_{t,\lambda}^{(1)}$, avoiding a biased solution due to the units of measurements of the response variable. Moreover, to sequentially ensure that $\gamma_t \in [0, \gamma_t^{(\max)}]$, the following rule must be adopted after the update of both γ_t and $\gamma_t^{(\max)}$,

$$\gamma_t = \max\{\min\{\gamma_t, \gamma_t^{(\max)}\}, 0\}.$$

Regarding the intercept, (4) is rewritten in terms of the elements in $\bar{\Pi}_{t,\lambda}$,

$$\hat{\beta}_t^{(0)}(\gamma_t) = \bar{\Pi}_{t,\lambda}^{(1)} - (\bar{\Pi}_{t,\lambda}^{(2)}, ..., \bar{\Pi}_{t,\lambda}^{(d+1)})' \hat{\beta}_t(\gamma_t), \tag{16}$$

where, different from (2), the subscript t denotes that the updating of $\hat{\beta}_t(\gamma_t)$ is sequential and $\bar{\Pi}_{t,\lambda}^{(j)}$ makes reference to the jth element of the vector.

D. Cyclic coordinate descent

Having defined the update of the penalty parameter, the next step is to calculate the values of $\hat{\beta}_t(\gamma_t)$ sequentially conditional to γ_t. Unfortunately, there is no analytical solution to find the minimum in (2). Several approaches are already available to optimise such problems, as pointed out in [27], [30], [31], when updating multiple regression weights sequentially, however the most appropriate is the Cyclical Coordinate Descent (CCD) algorithm [38], [39]. The main advantage of CCD is computational efficiency, exploring an analytic expression for (2) taking into account a partial optimum conditional to one specific weight, while the others remain fixed. Hence the name *cycles*, because the analytical expression is considered to update all the weights successively until convergence.

As described by [27, Sec. 2.4.2], the algorithm will propose an arbitrary order for the predictors and cycle trough them. In RAC, CCD is adopted to update the regression weights using the columns of $\Sigma_{t,\lambda}$. The derived formula (17) is a straightforward adaption of the estimation procedure when CCD is used to estimate the coefficients of the batch LASSO. In streaming data we receive only one data point at a time, hence we estimate the coefficients using the adaptive estimate of the covariance matrix $\Sigma_{t,\lambda}$.

At each step j the weights $\hat{\beta}_t^{(j)}(\gamma_t)$ are updated by minimizing the analytic expression for (2) in this coordinate, maintaining the values of remaining variables $\beta_t^{(l)}(\gamma_t)$, $l \neq j$ fixed. To remove the effect of the other variables, CCD makes use of a partial residual $r_t^{(\cdot,j)} = \Sigma_{t,\lambda}^{(\cdot,1)} - \sum_{l \neq j} \Sigma_{t,\lambda}^{(\cdot,l)} \hat{\beta}_t^{(l)}(\gamma_t)$. Hence, the update is given by

$$\hat{\beta}_t^{(j)}(\gamma_t) \leftarrow S\left(\hat{\beta}_t^{(j)}(\gamma_t) + \left\langle \Sigma_{t,\lambda}^{(\cdot,j)}, r_t^{(\cdot,j)} \right\rangle, \gamma_t\right), \qquad (17)$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product and $S(\cdot)$ makes reference to the soft threshold operator defined by

$$\begin{aligned} S(z,\gamma) &= \text{sign}(z)(|z| - \gamma)_+ \\ &= \begin{cases} z - \gamma, & \text{if } z > 0 \text{ and } \gamma < |z| \\ z + \gamma, & \text{if } z < 0 \text{ and } \gamma < |z| \\ 0, & \text{if } \gamma > |z|, \end{cases} \end{aligned}$$

with $(\cdot)_+$ denoting the positive part.

Similar to the update proposed in [30] for the VAR coefficients, the calculations of $\hat{\beta}_t(\gamma_t)$ in RAC also involves a FF λ, used in (9) and (10).

E. Basis construction

At the core of our method is an autoregressive regression with basis designed to capture local structure, overlayed with LASSO-inspired complexity control to prevent overfitting. Of course, there are many approaches to constructing a suitable basis which embodies such features. The features of particularly interest relates to trend and simple curvature.

The two proposed specifications of basis X in this section are designed to capture the streaming local structure. Such bases try to mimic several well known time series unobserved components, namely trend and seasonality.

1) Trig basis: Consider the basis based on a Fourier coefficient expansion for $X = [\psi^{\sin} \ \psi^{\cos}]$, with

$$\psi^{cos} = \begin{bmatrix} | & | & \dots & | \\ \cos\omega_1 t & \cos\omega_2 t & & \cos\omega_j t \\ | & | & \dots & | \end{bmatrix} \quad \psi^{sin} = \begin{bmatrix} | & | & \dots & | \\ \sin\omega_1 t & \sin\omega_2 t & & \sin\omega_j t \\ | & | & \dots & | \end{bmatrix},$$

where the frequencies $\omega_1, ..., \omega_j$ are defined by the user. This is closely related to the so called ℓ_1 trend filtering from [40] which proposed a variation on the Hodrick-Prescott filter.

This basis is strongly recommended when the local structure of the stream has smooth curvatures, typical of a stationary periodic signal. For both simulations and real data application, the fixed quantities were defined as $\omega_j = e^{-2\pi} + 0.2(j-1)$ $\forall j \in \mathcal{J}$ as an attempt to mimic a Fourier transform. These choices were made based on the performance analysis in Section III.

2) Cycle basis: Define the vector

$$\Lambda_{(n,i)} := ((1/n)^i, (2/n)^i, ..., (n/n)^i)',$$

and for some $\upsilon \in \mathbb{R}^n$, denote the indefinite concatenation operator of the elements of υ by

$$\upsilon^R = (\upsilon_1, \upsilon_2, ..., \upsilon_n, \upsilon_1, \upsilon_2, ..., \upsilon_n, \upsilon_1, \upsilon_2 ...)'.$$

Let Ξ_{min} and Ξ_{max} denote the minimum and maximum sequence lengths, respectively. The basis can be defined as the concatenation of the vectors

$$X = [\Lambda^R_{(\Xi_{min},1)} \ \Lambda^R_{(\Xi_{min},2)} \ \dots \ \Lambda^R_{(\Xi_{max},1)} \ \Lambda^R_{(\Xi_{max},2)}]$$

where the number of columns in X is a function of the quantities Ξ_{min} and Ξ_{max}. The user needs to specify Ξ_{min} and Ξ_{max}, the minimum and maximum sequence lengths in the columns of X. To avoid scaling problems, the jth column $X^{(\cdot,j)}$ is scaled by its maximum value. After this scaling, the X matrix contains in the first column the sequence from 1 to Ξ_{min}, the second column is the square of the first, and so on until the last sequence from 1 to Ξ_{max} and its square. In the following sections, we set the hyperparameters as $\Xi_{min} = 6$ and $\Xi_{max} = 40$. Such choices were also based on performance in simulation from Section III. In practice we may set the latter to a large value based on available processing resources.

F. Anomaly detection using a weighted sum of chi-squared random variables

To perform anomaly detection with a conditional model such as the RAC, the most straightforward approach is to look for anomalous values in the residuals $\hat{\varepsilon}_1, \hat{\varepsilon}_2, ...$ at each tick t. Intuitively, if the stream is well behaved, the model should be able to fit the local structure of $y_1, y_2, ...$, which will result in residuals close to zero. On the other hand, if the stream is poorly behaved, the residuals will exhibit anomalous values.

After estimation of $\hat{\beta}_t(\gamma_t)$ and the actual value y_{t+1} is observed, the residual one-step ahead forecasting error is

$$\hat{\varepsilon}_{t+1} = y_{t+1} - \hat{y}_{t+1}$$

where $\hat{y}_{t+1} = x'_{t+1}\hat{\beta}_t(\gamma_t)$ denotes the one-step ahead forecast. In the streaming context this generates an unending sequence of residuals, upon which we will make inference on

$$\xi_{t+1} = \left(\frac{y_{t+1} - \hat{y}_{t+1}}{\sqrt{\phi_{t+1}}} \right)^2,$$

where ϕ_t is a scalar value associated with the variance of y_t, i.e., $\phi_{t+1} = \Sigma^{(1,1)}_{t+1,\lambda}$ matrix updated by (10). Assuming *i.i.d.* data, this standardised quantity behaves as

$$\left(\frac{y_{t+1} - \hat{y}_{t+1}}{\sqrt{\phi_t}} \right) \sim N(0,1) \implies \xi_{t+1} \sim \chi^2_1.$$

This follows by assumption of *i.i.d.* normally distributed error. This is a strong parametric assumption typical of much of the change-point detection literature. Of course, if the assumption is invalid the results of the algorithm will be questionable. We prefer this approach because of the computational demands of streaming data, which often preclude non-parametric approaches.

Since we are tracking a moving target it is convenient to estimate the squared residual sequence using the forgetting mechanism. This also provides a better means for calibration of the residuals, by computing

$$\kappa_{t,\theta} = \theta\kappa_{t-1,\theta} + \xi_t \tag{18}$$

$$\nu_{t,\theta} = \theta\nu_{t-1,\theta} + 1 \tag{19}$$

$$\bar{\xi}_{t,\theta} = \frac{\kappa_{t,\theta}}{\nu_{t,\theta}}, \tag{20}$$

fixing $\kappa_{0,\theta} = \kappa_{1,\theta} = \nu_{0,\theta} = \nu_{1,\theta} = 0$ and using the quantity $\bar{\xi}_{t,\theta}$ as the object for inference. Of course, this is the same recursive formulation as (6) - (8). The random formulation of this quantity is a weighted mixture of chi-square distributions, described by [34], which does not have closed form but can be well approximated in streaming contexts by the Hall-Buckley-Eagleson method (HBE) [41]. Using such result, a sequential anomaly detector is constructed using the HBE method. The user here needs to control the value of θ, which states how many observations will be averaged to detect a change at each tick t. This is essentially a second stage of smoothing, using a fixed FF in (18)-(20). Our experiments suggest that performance is robust for $0.9 < \theta < 1$.

To evaluate if there is an anomaly at each tick t, the p-value is computed as

$$p_t = F_{HBE}(\bar{\xi}_{t,\theta}, \xi_t)$$

where $F_{HBE}(\cdot)$ is the cumulative distribution function (cdf) of a positively-weighted sum of chi-squared random variables using the HBE method with coefficient vector $\bar{\xi}_{t,\theta}$ evaluated at quantile ξ_t. Considering a significance level α, a change at tick t is flagged if $p_t \leq \alpha$.

G. Algorithm

An illustration on how RAC is sequentially performed is presented in Algorithm 1.

Algorithm 1 **R**eal-Time **A**daptive **C**omponent

Require: $\lambda \in (0.6, 1]$, η_γ, θ, $\Pi_{0,\lambda}$, $\Sigma_{0,\lambda}$, $w_{0,\lambda}$, $\hat{\beta}_0(\gamma_t)$, γ_0, $\kappa_{0,\theta}$, $\nu_{0,\theta}$, $\mathcal{B}$

for $t \leftarrow 1, 2, \ldots$ **do**
  Receive $(y_t, x_t^{(1)}, \ldots, x_t^{(d)})$
  Update $w_{t,\lambda}$, $\bar{\Pi}_{t,\lambda}$, $\Sigma_{t,\lambda}$, $\hat{\beta}_t^{(0)}(\gamma_t)$, $\gamma_t^{(max)}$, $\bar{\xi}_{t,\theta}$
  Estimate $\hat{\beta}_t(\gamma_t)$ using $\Sigma_{t,\lambda}$ and γ_t
  Update γ_t with $\gamma_t = \max\{\min\{\gamma_t, \gamma_t^{(\max)}\}, 0\}$
  if $t \geq \mathcal{B}$ **then**
    $p_t = F_{HBE}(\bar{\xi}_{t,\theta}, \xi_t)$

H. Computational considerations

As noted in [31], the biggest cost of this method is the calculation of the derivative in (14), which involves the inversion of the sample covariance matrix. In the same work, the authors proved that an approximation to the derivative (14) is given by

$$\frac{\partial\hat{\beta}_t(\gamma_t)}{\partial\gamma_t} \approx -(diag(\Sigma_t))^{-1} sign(\hat{\beta}_t(\gamma_t))$$

where $diag(\Sigma_t)$ represents the diagonal elements of the matrix. This approximation has time and memory demand proportional to the cardinality of $\mathcal{A}_t(\gamma_t)$. The compute time of RAC was evaluated using 100 Monte Carlo replicates of $N(0,1)$ of size 50000. Table I illustrates computational times when varying the number of variables. Results were evaluated using a personal computer with Intel Core i5, 1.8 GHz, 8 GB.

TABLE I
AVERAGE COMPUTE TIME WHEN RUNNING RAC.

Number of variables	5	10	50	100
Time (seconds)	99.2	163.5	1148.1	2463.9

III. SIMULATION STUDY

In this section we assess the performance of RAC in two respects, *estimation* and *detection*. The first is concerned with the method's forecasting ability to track an arbitrary signal with underlying structure varying over time. The second is concerned with the detection capabilities of the method when facing a change in the signal's underlying proprieties. RAC will be compared to other methods discussed in the literature, such as AFF [15], online VAR[1] [30], CUSUM [42], EWMA [43], probabilistic EWMA (PEWMA) [44], covariate shift detection with EWMA (SD-EWMA) [45], two-stage covariate shift detection with EWMA (TSSD-EWMA)[45], conformal prediction with k-nearest neighbours (KNN-CAD) [46] and Seasonal Hybrid ESD (S-H-ESD) [47] which uses the Generalized Extreme Studentized Deviate test to detect anomalies [48]. A simple benchmark, which we will refer to as the NAIVE benchmark is based on using y_t as the forecast $\hat{y}_{t+1}$ is also included. Since online VAR and NAIVE do not

[1]Note that here, as we are only considering one stream at a time, the VAR model is actually an univariate autoregressive model.

have a proper detection method, they are not included when evaluating detection performance.

Our experiments are based on the data generated by the following process

$$y_t = \begin{cases} 2\cos\left(2\pi\frac{1}{100}t + 0.1\right) + \varepsilon_t, & \text{if } t \leq 5000 \\ 10\cos\left(2\pi\frac{1}{200}t + 0.1\right) + \varepsilon_t, & \text{if } 5001 \leq t \leq 10000 \end{cases} \tag{21}$$

where $\varepsilon_t \sim N(0,1)$. The results obtained in Sections III-A and III-B are based on adopting $\mathcal{B} = 1000$ observations, $\eta_\gamma \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$ and $\lambda \in \{0.6, 0.7, 0.8, 0.9, 0.95, 0.995\}$.

A. Estimation performance

A straightforward way to evaluate prediction performance is averaging forecasting error metrics, such as Mean Square Error (MSE) and Mean Absolute Error (MAE), over the 100 replicates for all methods. These metrics are defined as

$$MSE = \sum_{t=\mathcal{B}+1}^{T} \frac{1}{T-\mathcal{B}}(\hat{y}_{t+1} - y_{t+1})^2, \tag{22}$$

$$MAE = \sum_{t=\mathcal{B}+1}^{T} \frac{1}{T-\mathcal{B}}|\hat{y}_{t+1} - y_{t+1}|, \tag{23}$$

where T is the total length of the stream.

The averaged forecasting metrics are displayed in Table II. It is clear that RAC can provide good forecasting performance provided the control parameters are selected appropriately. Notably, both NAIVE benchmark and AFF have reasonably good performances. However, their performance for anomaly detection is low, pointing to the dichotomy between forecasting accuracy and anomaly detection capability. Note that for the online VAR model of [30], computation became infeasible for $\lambda < 0.7$. Also, MSE and MAE for AFF and NAIVE are displayed in only one column. Regarding AFF, the analyst just chooses η_λ, the step of gradient descent, to update the FF λ_t, while NAIVE no choices are needed. Figures 1 and 2 show the averages of penalty term γ_t, p-value and number of non zero coefficients for $\hat{\beta}(\gamma_t)$ using *Cycle* and *Trig* basis respectively. These plots illustrates the combination of η_γ and λ that minimises the mean square error (MSE) of one step ahead forecasting error.

Some observations about Figures 1 and 2. First, RAC is able to continuously track an arbitrary target after its latent underlying structure faces a change. Second, around the change point, $\tau^* = 5000$, the average penalty parameter, γ_t, increases as the average number of non zero weights decreases. This is expected, because after the change point, RAC will re-estimate new weights to filter the new underlying structure of the signal. This is consistent with the findings of [31] that the optimal regularization parameter is time-varying when the underlying distribution is nonstationary. Also, we note in Figures 1 and 2 the average behaviour of the p-values around the change point. Considering a 5% significance level, on average RAC rejects the null hypothesis that the stream is not experiencing anomalous behaviour around $t = \tau^*$. Finally, regarding the average non zero weights plots, a *LOESS*-curve [49] is fitted to provide additional intuition. For both bases, on average, the number of non zero weights in each segment appears reasonably stable, though subject to substantial variability. The *Trig* basis, on average, seems better suited to this specific signal. A possible explanation is that with the *Cycle* basis, RAC selects quadratic terms to fit local curvatures. This is evident after $t = 5000$, when the amplitude of the curve is higher, fewer quadratic terms are selected – implying the slight decay in the *LOESS* curve.

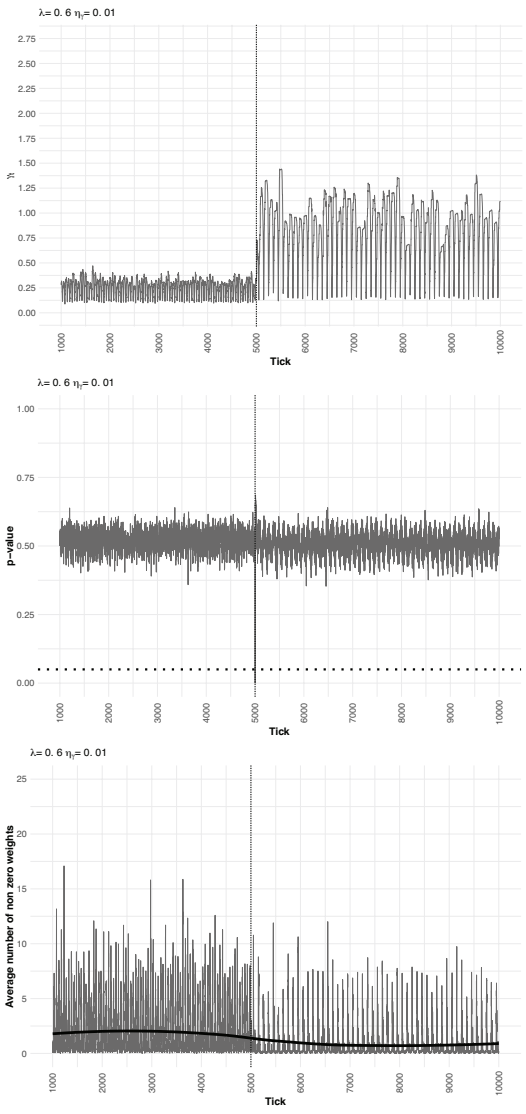

Fig. 1. Average quantities of the penalty term γ_t, p-value with a dashed horizontal line at the 5% significance level and number of non zero weights $\hat{\beta}(\gamma_t)$ with a smoothed *LOESS* curve over 100 Monte Carlo simulations using *Cycle* basis. The title of the figures shows the parameters η_γ and λ that produces the minimum Mean Square Error.

B. Detection performance

The AFF [15] method has its own detection method while RAC will use the method proposed in Section II-F. Other methods in the comparative study use the detection method as described in the corresponding papers (see the introduction of Section III). Allowing ϑ observations after a change is a

TABLE II
AVERAGE FORECASTING ACCURACY MEASURES, OVER 100 REPLICATES OF (21). NOTE THAT THE RESULTS OF AFF ARE THE SAME ACROSS COLUMNS BECAUSE THE FORGETTING FACTORS ARE ADAPTIVE. ALSO, THE ONLINE VAR MODEL FOR VALUES OF $\lambda < 0.7$ WERE NOT FEASIBLE TO COMPUTE.

			MSE							MAE						
Basis	Method	(η_γ, λ)	0.6	0.7	0.8	0.9	0.95	0.995	-	0.6	0.7	0.8	0.9	0.95	0.995	-
-	AFF	10^{-1}	-	-	-	-	-	-	1.62	-	-	-	-	-	-	1.01
		10^{-2}	-	-	-	-	-	-	1.45	-	-	-	-	-	-	0.96
		10^{-3}	-	-	-	-	-	-	1.44	-	-	-	-	-	-	0.95
		10^{-4}	-	-	-	-	-	-	1.46	-	-	-	-	-	-	0.96
		10^{-5}	-	-	-	-	-	-	1.89	-	-	-	-	-	-	1.10
		10^{-6}	-	-	-	-	-	-	3.47	-	-	-	-	-	-	1.44
	NAIVE	-	-	-	-	-	-	-	2.03	-	-	-	-	-	-	1.13
Cycle	RAC	10^{-1}	0.75	1.20	3.29	6.31	14.81	33.88	-	0.61	0.75	1.01	1.60	2.65	4.45	-
		10^{-2}	0.73	1.12	2.68	5.97	14.39	33.35	-	0.61	0.74	0.98	1.58	2.62	4.41	-
		10^{-3}	0.76	1.11	2.29	5.65	13.86	31.02	-	0.63	0.75	0.98	1.56	2.60	4.27	-
		10^{-4}	0.79	1.11	2.23	7.31	14.82	30.74	-	0.67	0.80	1.05	1.68	2.67	4.26	-
		10^{-5}	0.97	1.38	2.69	8.64	21.77	31.04	-	0.75	0.90	1.19	1.87	3.10	4.28	-
		10^{-6}	1.21	1.82	3.37	12.52	32.76	35.15	-	0.84	1.03	1.32	2.24	3.74	4.56	-
	VAR	-	-	11.54	9.21	8.29	8.27	8.23	-	-	1.88	1.89	1.78	1.97	1.80	-
Trig	RAC	10^{-1}	0.65	1.08	1.86	3.23	7.25	8.05	-	0.61	0.75	0.97	1.43	2.14	2.16	-
		10^{-2}	0.63	0.96	1.53	3.10	7.21	8.02	-	0.61	0.74	0.92	1.41	2.13	2.15	-
		10^{-3}	0.64	0.91	1.30	2.74	6.56	8.12	-	0.63	0.75	0.90	1.32	2.01	2.17	-
		10^{-4}	0.68	0.98	1.34	1.89	3.74	8.74	-	0.66	0.79	0.92	1.08	1.45	2.24	-
		10^{-5}	0.75	1.08	1.51	2.00	2.28	7.94	-	0.69	0.83	0.97	1.11	1.17	2.11	-
		10^{-6}	0.83	1.19	1.78	2.49	2.14	6.26	-	0.72	0.87	1.05	1.24	1.15	1.87	-
	VAR	-	-	11.67	9.02	8.78	8.30	8.02	-	-	1.84	1.86	2.23	1.79	1.74	-

common practice [50], and also adopted in this work, since there may be a slight delay after a change occurs. Considering 100 replicates of (21), detection performance is evaluated as:

- For each replicate, false detection is calculated as the average number of events that are defined as anomalous when $y_t \notin (\tau^*, \tau^* + \vartheta)$;
- For each replicate, correct detection is a indicator function that assumes 1 if an anomaly is detected when $y_t \in (\tau^*, \tau^* + \vartheta)$ and zero otherwise.

Table III show the average number of False Detection (FD) and average Correct Detection (CD) rate over 100 when adopting $\vartheta = 20$ for both basis *Cycle* and *Trig*. Parameters for RAC were fixed at $\alpha = 0.01$ and $\theta = 0.95$ and only those results for parameter configurations rendering CD equal to 1 and FD less than 20 were displayed. Although only $\alpha = 0.01$ is reported here, more simulations are available upon request. Regarding parameter setting of competing methods, Table III depicts the parameter values with the corresponding notation of the R packages (*otsda*, *AnomalyDetection* and *ffstream*). For PEWMA, SD-EWMA, TSSD-EWMA and KNN-CAD the training set is fixed to 1000 data points. In general we have selected the default parameters.

The main points from Table III; first, RAC appears accurate at detecting anomalies due to its high CD rates, and corresponding low FD rates; second the performance of RAC is somehow dependent on the choice of control parameters; third in this simulation both choices of basis perform comparably. Finally, the effect of λ has a different impact regarding estimation and detection performance, where larger λ is associated with better detection performance.

The poor CD performance of AFF is due to the fact that it cannot cope with underlying structure in the data. It was designed to calculate a dynamic average under the assumption of *i.i.d.* Gaussian observations. If seasonality is present in the data, filtering the unconditional mean in this case will end up resulting in *p*-values oscillating with the stream's unconditional mean.

TABLE III
AVERAGE OF CORRECT (CD) AND FALSE DETECTION (FD) OVER 100 REPLICATES OF THE PROCESS (21). STANDARD DEVIATIONS ARE IN PARENTHESIS.

RAC (Basis, λ, η_γ)	CD	FD
RAC (Cycle, 0.995, 10^{-1})	1.00 (0.00)	2.03 (1.41)
RAC (Cycle, 0.995, 10^{-2})	1.00 (0.00)	1.62 (1.30)
RAC (Cycle, 0.995, 10^{-3})	1.00 (0.00)	1.57 (1.40)
RAC (Cycle, 0.995, 10^{-4})	1.00 (0.00)	1.72 (1.23)
RAC (Cycle, 0.995, 10^{-5})	1.00 (0.00)	1.65 (1.14)
RAC (Cycle, 0.995, 10^{-6})	1.00 (0.00)	1.96 (1.23)
RAC (Trig, 0.995, 10^{-1})	1.00 (0.00)	9.63 (3.06)
RAC (Trig, 0.995, 10^{-2})	1.00 (0.00)	10.31 (2.64)
RAC (Trig, 0.995, 10^{-3})	1.00 (0.00)	11.73 (3.04)
RAC (Trig, 0.995, 10^{-4})	1.00 (0.00)	12.25 (3.23)
RAC (Trig, 0.995, 10^{-5})	1.00 (0.00)	12.08 (3.53)
RAC (Trig, 0.995, 10^{-6})	1.00 (0.00)	11.92 (3.24)
Model	**CD**	**FD**
PEWMA (*alpha0* = 0.8, *beta* = 0.1, *l* = 3)	1.00 (0.00)	63.37 (7.70)
SD-EWMA (*threshold* = 0.01, *l* = 3)	1.00 (0.00)	22.83 (4.27)
TSSD-EWMA (*threshold* = 0.01, *l* = 3, *m* = 5)	0.11 (0.31)	0.02 (0.14)
KNN-CAD (*threshold* = 1, *l* = 19, *k* = 27)	0.82 (0.38)	3.13 (1.23)
S-H-ESD (*max_anoms* = 0.1, *period* = 100)	0.05 (0.22)	880.26 (30.39)
AFF (*eta* = 10^{-6}, *BL* = 1000)	0.42 (0.49)	9.82 (0.73)
CUSUM (*k* = 0.55, *h* = 0.5)	0.99 (0.04)	18.00 (0.04)
EWMA (*r* = 1, *L* = 3.09)	0.95 (0.20)	0.56 (0.75)

C. Detection performance on real data

In order to provide more insight concerning multiple change point detection performance, five time series from Twitter were used. RAC is compared with the state-of-the art methods discussed above. For comparison, as a scoring function, we adopt the Numenta Anomaly Benchmark (NAB) [51], which is an anomaly detector scoring function designed to compare different algorithms on streaming data.

The NAB scoring method operates on windows and penalizes false detections outside the anomaly window. Weights are given to True Positives (TP), False Positives (FP), True Negative (TN) and False Negative (FN). These windows are centered around the *ground truth* anomalies and the scoring function uses weights for TP, FP, TN and FN chosen by the user. In case of multiple detections within the same window, more credit is given to the earliest and counted as true positive,

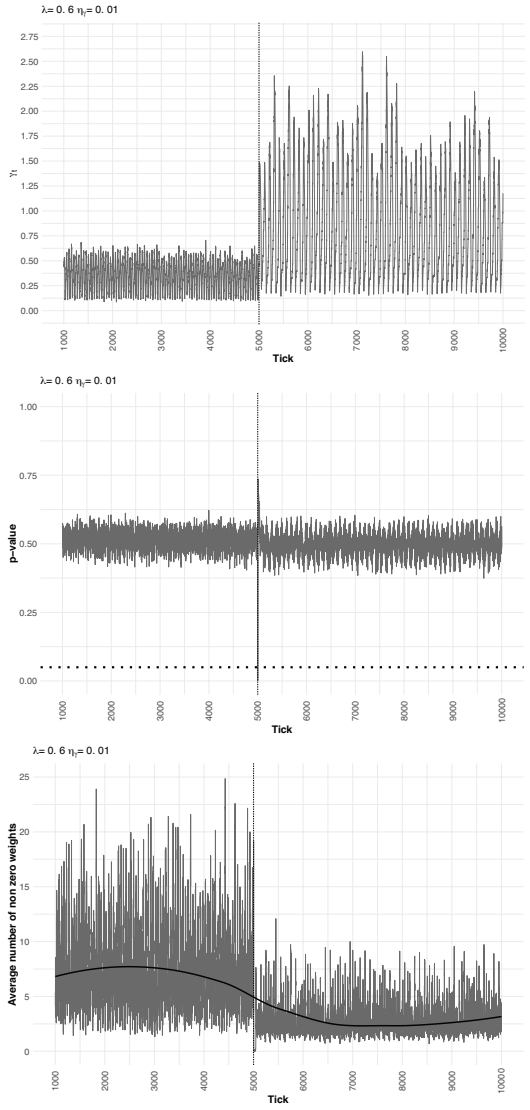

Fig. 2. Average quantities of the penalty term γ_t, p-value with a dashed horizontal line at the 5% significance level and number of non zero weights $\hat{\beta}(\gamma_t)$ with a smoothed *LOESS* curve over 100 Monte Carlo simulations using *Trig* basis. The title of the figures shows the parameters η_γ and λ that produces the minimum Mean Square Error.

while others are ignored. Details are given in [51] and results obtained here followed the code available in the *otsad* R package.

The analysed time series are collections of Twitter 'mentions' of traded tech companies namely, Apple, Amazon, Facebook, Google and IBM. Data sets are available at the NAB repository[2] and in the *otsad* R package. Each time series value reflects the number of mentions of each company every five minutes. Average normalized NAB scores (normalized by the perfect and null detectors' performance) are presented in Table IV. For illustration purposes only results with average normalized scores higher than 30 and positive scores of FP and FN are presented for RAC. Apart from the S-H-ESD parameters that followed the implementation available in the NAB repository, parameters for the competing methods were set as depicted in III.

[2]https://github.com/numenta/NAB

TABLE IV
AVERAGE OF NAB NORMALIZED FUNCTIONS OVER 5 TIME SERIES FROM TWITTER. THE MAXIMUM AND BEST SCORE IS 100.

Model (Basis, λ, η_γ)	NAB	Reward FP	Reward FN
RAC (Cycle, 0.7, 10^{-6})	49.88	37.04	56.03
RAC (Cycle, 0.9, 10^{-1})	31.45	16.75	37.08
RAC (Cycle, 0.9, 10^{-4})	40.75	28.29	46.05
RAC (Cycle, 0.9, 10^{-5})	40.88	29.26	46.14
RAC (Cycle, 0.9, 10^{-6})	51.96	37.08	58.53
RAC (Trig, 0.7, 10^{-6})	49.88	37.04	56.03
RAC (Trig, 0.8, 10^{-6})	33.92	24.34	38.17
RAC (Trig, 0.9, 10^{-1})	32.29	11.97	40.97
RAC (Trig, 0.9, 10^{-2})	43.42	24.52	51.17
Model	NAB	Reward FP	Reward FN
PEWMA	-570.77	-1241.19	-347.18
SD-EWMA	-297.04	-693.43	-164.69
TSSD-EWMA	20.54	11.31	24.25
KNN-CAD	62.33	33.73	73.22
S-H-ESD	49.09	14.24	62.17
AFF	8.70	-16.36	18.02
CUSUM	30.21	4.37	40.69
EWMA	25.90	8.91	32.82

The key finding from Table IV is that the performance achieved by RAC, although slightly better than S-H-ESD under suitable choice of parameters and worse than KNN-CAD, is a competitive detector regarding detection score, FP and FN. Although RAC generates a substantial amount of FP and FN (see second and third column of Table IV), this can be controlled by varying the significance level α. The poor performance of PEWMA and SD-EWMA on this experiment is due to the fact these two methods are made for stationary streams.

IV. REAL DATA: CYBER-SECURITY

Cyber-crime is an increasing burden to society, costing around $600 billion per annum [52]. Operational anomaly detection methods in this area predominantly rely on signature-based methods, which seek to identify events and behaviour of known form [53]. Such methods are incapable of dealing with so-called "zero-day" attacks – activities that have not previously been reported [54]. Moreover, the increasing intensity and consequences of cyber-attacks suggest that existing signature-based methods are insufficient. There is a burgeoning research area using automatically collected computer and network data in conjunction with statistical and machine learning methods which is focused on anomaly detection. Specifically, detecting departures from "normal" behaviour [55], [56]. These data-driven methods are intended to complement existing signature-based tools.

There are a number of challenges in developing statistical methods for cyber-security. Some relate to the volume and velocity of the data – typical data sources can be vast in a large enterprise, and as demonstrated later, data can arrive at very high rates. Another type of challenge relates to practical usage. Any data-driven anomaly detection method will suffer from false positives, and these create a misleading and potentially

costly false signal. In this section we deploy RAC against cyber-security data sets provided by Los Alamos National Laboratory (LANL). The data set is described in detail in [53] and is available online[3].

We focus on the *host Log* data, which comprises a subset of computer event logs collected from all computers running the Microsoft Windows operating system on LANL's enterprise network. Events from the *host log* included in the data set are all related to authentication and process activity on each machine. We focus on this data source, rather than the more widely studied *Network Flow* data because, as noted in [53], remote attackers and malicious insiders increasingly use encryption, hence reducing the effectiveness of communication-based detection mechanisms.

There are several kinds of events and log on-types described by [53]. We only consider those which are driven only by human behaviour, i.e., which do not run any automated process periodically. The selected events were

- 4624: An account was successfully logged on;
- 4625: An account failed to logon;
- 4634: An account was logged off;
- 4800: The workstation was locked;
- 4801: The workstation was unlocked;
- 4802: The screensaver was invoked;
- 4803: The screensaver was dismissed.

For each of these event types, we construct a data stream, $y_1, y_2, ...y_t, y_{t+1}, ...$, that counts the number of events per minute. For the LANL data, this yields seven time series, each consisting of 129600 data points. A one-minute window is a plausible size in this context, given constraints on data collection and the requirement for timely detection. We then want to assess RAC's performance, in terms of estimation and detection, compared to AFF, online VAR and NAIVE. For forecasting accuracy, similar to the simulations in Section III, MSE and MAE are adopted.

The streams are formed from count data, and hence strictly non-negative. RAC was not designed to specifically deal with this case – modifications are possible, though challenging and hence left to future work. Here, we use a simple rule whereby negative predictions from RAC are set to zero. While this is perhaps the crudest possible fix, it can be applied consistently without computational overhead.

To explore the capabilities of RAC, the results are considered in three different contexts:

- $\mathcal{W}$: The whole data set for each event will be used, i.e., $t = 1, ..., 129600$;
- $\mathcal{A}_t(\gamma_t) \neq \emptyset$: Only predictions from those points where at least one weight of $\hat{\beta}_t(\gamma_t)$ is non zero will be considered;
- $\mathcal{A}_t(\gamma_t) = \emptyset$: Only predictions from those points where all the weights of $\hat{\beta}_t(\gamma_t)$ are equals to zero will be considered.

The results for estimation metrics considering one step ahead forecasting error over the sets $\mathcal{W}$, $\mathcal{A}_t(\gamma_t) \neq \emptyset$ and $\mathcal{A}_t(\gamma_t) = \emptyset$ are depicted in Tables V, VI and VII. The initial values were fixed as follows. For burn-in, we use $\mathcal{B} = 1440$ observations, i.e., one day of data. The remaining control parameters are selected as $\gamma_0 = 10$, $\eta_\gamma = 0.01$, and $\lambda = 0.6$. The choices of η_γ and λ are the values that produce minimum MSE in the simulations of Section III, while γ_0 was fixed based on empirical evidence while running the simulations. Regarding online VAR, we fixed $\lambda = 0.995$ which gave the best result regarding MSE performance in simulations.

Considering the forecasting errors when $y_t \in \mathcal{W}$, displayed in Table V, RAC is able to accurately track a complicated target with underlying structure varying over time. Considering MSE and MAE, RAC outperform the other methods in all of the 7 cyber-security examples, regardless of the selected basis. The complex structure of RAC allows for different combination of basis functions, and notably allows for all regression weights to be forced to zero. This provides some modelling advantage, as the results in Table VI clearly illustrate. Such advantage takes into account, for the calculation of both forecasting metrics, only events where $\mathcal{A}_t(\gamma_t) \neq \emptyset$, i.e., ticks of y_t where RAC estimates weights $\hat{\beta}_t(\gamma_t) \neq 0$. In comparing bases, on average, the *Trig* basis tends to select more coefficients than the *Cycle* basis, producing smaller errors in $\mathcal{A}_t(\gamma_t) \neq \emptyset$. This can be verified in the last two columns of Table VI, the total number of points in which the weights $\hat{\beta}_t(\gamma_t)$ are different from zero and the proportion compared to the total number of observations $T = 129000$.

Finally, consider $\mathcal{A}_t(\gamma_t) = \emptyset$, the case in which all RAC weights $\hat{\beta}_t(\gamma_t)$ equal zero. In this case, RAC outperformed the benchmarks in terms of forecasting performance. This suggests that RAC is capable of handling periods that are locally constant, in addition to periods exhibiting high non-linearity. This also provides evidence about the proposed updates given by (15) and (16), since the forecast is produced only by the dynamic intercept $\hat{\beta}_t^{(0)}(\gamma_t)$ in this set.

TABLE V
ONE STEP AHEAD ESTIMATION ACCURACY IN CYBER-SECURITY EVENTS BY ID USING THE WHOLE SET $y_t \in \mathcal{W}$.

Basis	Error Metric	Event	AFF	VAR	NAIVE	RAC
Cycle	MSE	4624	103.38	113.58	249.88	62.65
		4625	2.12	1.84	4.65	1.13
		4634	102.92	113.22	249.13	62.49
		4800	15.75	19.73	40.77	10.67
		4801	15.24	17.06	37.81	9.77
		4802	5.54	5.90	12.83	3.47
		4803	22.97	21.84	46.71	14.57
	MAE	4624	4.81	5.13	7.60	3.76
		4625	0.69	0.65	1.00	0.50
		4634	4.80	5.13	7.60	3.76
		4800	1.96	2.23	3.15	1.59
		4801	1.93	2.09	3.05	1.52
		4802	1.31	1.34	1.96	1.00
		4803	1.39	1.36	2.01	1.02
Trig	MSE	4624	103.38	113.17	249.88	61.59
		4625	2.12	3.17	4.65	1.10
		4634	102.92	112.86	249.13	61.99
		4800	15.75	20.30	40.77	10.84
		4801	15.24	17.53	37.81	9.19
		4802	5.54	8.05	12.83	3.26
		4803	22.97	25.84	46.71	13.92
	MAE	4624	4.81	5.22	7.60	3.75
		4625	0.69	0.85	1.00	0.50
		4634	4.80	5.22	7.60	3.75
		4800	1.96	2.31	3.15	1.59
		4801	1.93	2.16	3.05	1.51
		4802	1.31	1.60	1.96	0.99
		4803	1.39	1.63	2.01	1.01

It is nearly impossible to evaluate detection performance on real cyber-security data because of the paucity of labels, i.e., there is no information regarding when, or if, an anomaly happened [57]. We perform anomaly detection using the

[3]https://csr.lanl.gov/data/2017.html

TABLE VI
ONE STEP AHEAD ESTIMATION ACCURACY IN CYBER-SECURITY EVENTS BY ID ADOPTING $\mathcal{A}_t(\gamma_t) \neq \emptyset$.

Basis	Error Metric	Event	AFF	VAR	NAIVE	RAC	Size	Prop
Cycle	MSE	4624	149.68	172.08	238.24	101.11	26251	20.26%
		4625	6.00	5.30	8.53	3.65	12408	9.57%
		4634	147.04	170.03	235.85	100.47	26108	20.15%
		4800	19.38	25.15	34.90	16.28	26111	20.15%
		4801	18.89	21.94	30.67	14.44	26249	20.25%
		4802	8.95	10.16	14.13	6.89	28277	21.82%
		4803	61.49	51.56	135.17	44.42	27721	21.39%
	MAE	4624	4.22	4.43	5.11	3.58	26251	20.26%
		4625	1.44	1.39	1.60	1.19	12408	9.57%
		4634	4.14	4.35	5.03	3.53	26108	20.15%
		4800	1.66	1.87	2.05	1.47	26111	20.15%
		4801	1.68	1.83	2.04	1.45	26249	20.25%
		4802	1.38	1.41	1.59	1.14	28277	21.82%
		4803	1.50	1.50	1.78	1.23	27721	21.39%
Trig	MSE	4624	57.68	66.80	99.75	39.49	51842	40.00%
		4625	0.91	1.18	1.59	0.53	70446	54.36%
		4634	57.06	66.41	101.59	40.71	51798	39.97%
		4800	7.38	10.31	17.04	7.09	51580	39.80%
		4801	6.51	8.12	13.05	4.64	51966	40.10%
		4802	3.84	4.93	6.83	2.55	56723	43.77%
		4803	30.30	30.07	38.37	14.58	54464	42.02%
	MAE	4624	2.03	2.33	2.83	1.74	51842	40.00%
		4625	0.29	0.34	0.38	0.23	70446	54.36%
		4634	2.08	2.37	2.89	1.79	51798	39.97%
		4800	0.93	1.12	1.32	0.83	51580	39.80%
		4801	0.87	1.00	1.21	0.75	51966	40.10%
		4802	0.84	0.93	1.08	0.66	56723	43.77%
		4803	0.90	0.98	1.15	0.70	54464	42.02%

TABLE VII
ONE STEP AHEAD ESTIMATION ACCURACY IN CYBERSECURITY EVENTS BY ID ADOPTING $\mathcal{A}_t(\gamma_t) = \emptyset$.

Basis	Error Metric	Event	AFF	VAR	NAIVE	RAC
Cycle	MSE	4624	91.04	98.51	251.99	52.63
		4625	1.71	1.47	4.23	0.87
		4634	91.24	98.69	251.67	52.75
		4800	14.72	18.35	42.06	9.21
		4801	14.22	15.80	39.48	8.55
		4802	4.56	4.69	12.41	2.49
		4803	12.21	13.63	22.09	6.27
	MAE	4624	4.95	5.31	8.23	3.80
		4625	0.61	0.57	0.93	0.43
		4634	4.96	5.32	8.24	3.82
		4800	2.03	2.33	3.42	1.62
		4801	1.99	2.16	3.31	1.54
		4802	1.29	1.33	2.06	0.96
		4803	1.35	1.32	2.06	0.96
Trig	MSE	4624	133.09	144.66	348.84	75.99
		4625	3.55	5.60	8.29	1.78
		4634	132.72	144.36	346.28	75.82
		4800	21.14	27.03	56.21	13.24
		4801	20.96	23.95	54.20	12.19
		4802	6.82	10.53	17.42	3.78
		4803	17.27	22.72	52.01	13.21
	MAE	4624	6.64	7.18	10.77	5.07
		4625	1.16	1.47	1.73	0.82
		4634	6.61	7.16	10.72	5.04
		4800	2.63	3.11	4.34	2.09
		4801	2.63	2.95	4.28	2.02
		4802	1.68	2.13	2.63	1.24
		4803	1.73	2.10	2.63	1.24

approach described in Section II-F, fixing a FF $\theta = 0.95$ and $\alpha = 1/\mathcal{B}$, i.e., we intend to capture 1 anomaly per day. Figure 3 show six days of the first week of LANL's *host log* data set for one of the described events (4803), excluding the first day, which was used as burn-in. The vertical lines make reference to detected anomalies. Some of them seem to capture unusual patterns, but as argued we cannot prove these are actual anomalies due to the absence of labelled data.

V. CONCLUSION

This paper propose RAC, a framework to estimate, forecast and perform anomaly detection in streaming data environments. Empirical results regarding estimation and detection performance, both for simulated and real data sets, provides evidence that our proposed framework is able to track a moving target, and identify changes in local structure.

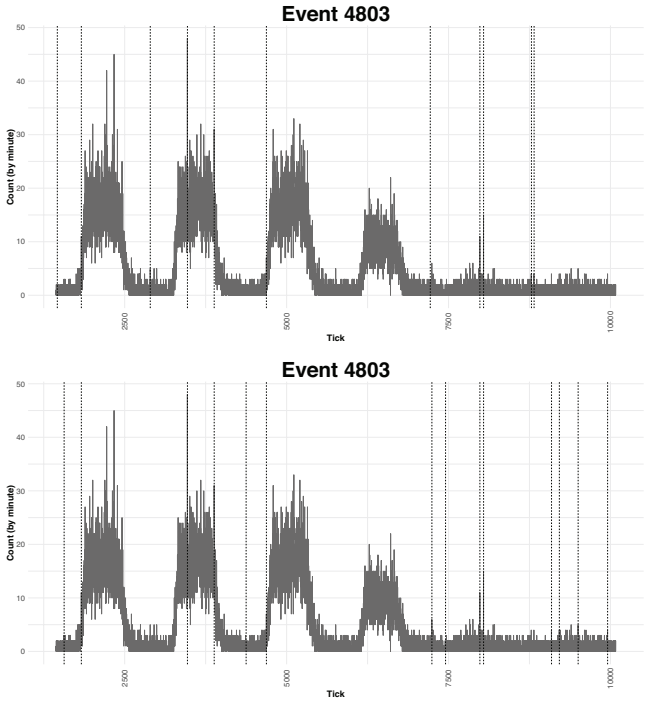

Fig. 3. RAC detection performance using *Cycle basis* (top) and *Trig basis* (bottom) on the event 4803. Vertical dashed lines are flagged change points, i.e., $p_t < 1/\mathcal{B}$.

In the cyber-security example, the forecasting accuracy of RAC was better than existing methods for almost all of the ID events. Additionally, both bases proposed to sequentially fit the underlying structure of the signal, namely *Trig* and *Cycle* had excellent performance. Regarding the detection performance, we could only evaluate it in simulation studies, due to the fact that LANL's cyber-security data set is unlabelled. Still, RAC performed well when comparing correct and false detection rates against the benchmarks.

For future work, we mention a few possible extensions of this framework. Firstly, sequential prediction intervals might be calculated adapting the results of [58] to a time-varying penalty term γ_t. Secondly, extensions respecting the range of response, essentially extending the framework to the coverage of GLMs.

REFERENCES

[1] C. C. Aggarwal, *Data streams: Models and Algorithms*. Springer, 2007, vol. 31.
[2] J. Gama, *Knowledge discovery from data streams*. CRC Press, 2010.
[3] N. A. Heard, D. J. Weston, K. Platanioti, D. J. Hand *et al.*, "Bayesian anomaly detection methods for social networks," *The Annals of Applied Statistics*, vol. 4, no. 2, pp. 645–662, 2010.
[4] M. Sayed-Mouchaweh and E. Lughofer, *Learning in non-stationary environments: methods and applications*. Springer, 2012.
[5] E. Lughofer and M. Sayed-Mouchaweh, *Predictive Maintenance in Dynamic Systems: Advanced Methods, Decision Support Tools and Real-World Applications*. Springer, 2019.
[6] W. K. Härdle, W. Wang, and L. Yu, "Tenet: Tail-event driven network risk," *Journal of Econometrics*, vol. 192, no. 2, pp. 499–513, 2016.
[7] D. J. Weston, D. J. Hand, N. M. Adams, C. Whitrow, and P. Juszczak, "Plastic card fraud detection using peer group analysis," *Advances in Data Analysis and Classification*, vol. 2, no. 1, pp. 45–62, 2008.

[8] F. D.-H. Lau, L. J. Butler, N. M. Adams, M. Z. Elshafie, and M. A. Girolami, "Real-time statistical modelling of data generated from self-sensing bridges," *Proceedings of the Institution of Civil Engineers-Smart Infrastructure and Construction*, pp. 1–42, 2018.

[9] A. A. Benczúr, L. Kocsis, and R. Pálovics, "Online machine learning in big data streams," *arXiv preprint arXiv:1802.05872*, 2018.

[10] C. Fahy, S. Yang, and M. Gongora, "Ant colony stream clustering: A fast density clustering algorithm for dynamic data streams," *IEEE Transactions on Cybernetics*, vol. 49, no. 6, pp. 2215–2228, 2019.

[11] C. Fahy and S. Yang, "Finding and tracking multi-density clusters in online dynamic data streams," *IEEE Transactions on Big Data*, pp. 1–1, 2019.

[12] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[13] I. Khamassi, M. Sayed-Mouchaweh, M. Hammami, and K. Ghédira, "Discussion and review on evolving data streams and concept drift adapting," *Evolving systems*, vol. 9, no. 1, pp. 1–23, 2018.

[14] J. Law and D. J. Wilkinson, "Composable models for online bayesian analysis of streaming data," *Statistics and Computing*, vol. 28, no. 6, pp. 1119–1137, 2018.

[15] D. A. Bodenham and N. M. Adams, "Continuous monitoring for changepoints in data streams using adaptive estimation," *Statistics and Computing*, vol. 27, no. 5, pp. 1257–1270, 2017.

[16] S. S. Haykin, *Adaptive filter theory*. Pearson Education India, 2008.

[17] C. Anagnostopoulos, D. K. Tasoulis, N. M. Adams, N. G. Pavlidis, and D. J. Hand, "Online linear and quadratic discriminant analysis with adaptive forgetting for streaming classification," *Statistical Analysis and Data Mining*, vol. 5, no. 2, pp. 139–166, 2012.

[18] A. Shaker and E. Lughofer, "Self-adaptive and local strategies for a smooth treatment of drifts in data streams," *Evolving Systems*, vol. 5, no. 4, pp. 239–257, 2014.

[19] M. Pratama, J. Lu, E. Lughofer, G. Zhang, and M. J. Er, "An incremental learning of concept drifts using evolving type-2 recurrent fuzzy neural networks," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 5, pp. 1175–1192, 2016.

[20] J. Steinhardt, S. Wager, and P. Liang, "The statistics of streaming sparse regression," *arXiv preprint arXiv:1412.4182*, 2014.

[21] M. Xu, M. Han, T. Qiu, and H. Lin, "Hybrid regularized echo state network for multivariate chaotic time series prediction," *IEEE Transactions on Cybernetics*, vol. 49, no. 6, pp. 2305–2315, 2019.

[22] P. Jing, Y. Su, X. Jin, and C. Zhang, "High-order temporal correlation model learning for time-series prediction," *IEEE Transactions on Cybernetics*, vol. 49, no. 6, pp. 2385–2397, 2019.

[23] H. ElMoaqet, D. M. Tilbury, and S. K. Ramachandran, "Multi-step ahead predictions for critical levels in physiological time series," *IEEE Transactions on Cybernetics*, vol. 46, no. 7, pp. 1704–1714, 2016.

[24] F. Rasheed and R. Alhajj, "A framework for periodic outlier pattern detection in time-series sequences," *IEEE Transactions on Cybernetics*, vol. 44, no. 5, pp. 569–582, 2014.

[25] A. Durand, O.-A. Maillard, and J. Pineau, "Streaming kernel regression with provably adaptive mean, variance, and regularization," *Journal of Machine Learning Research*, vol. 19, no. 1, pp. 650–683, 2018.

[26] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[27] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical learning with sparsity: the lasso and generalizations*. CRC press, 2015.

[28] N. Meinshausen, B. Yu *et al.*, "Lasso-type recovery of sparse representations for high-dimensional data," *The Annals of Statistics*, vol. 37, no. 1, pp. 246–270, 2009.

[29] S. A. Van de Geer *et al.*, "High-dimensional generalized linear models and the lasso," *The Annals of Statistics*, vol. 36, no. 2, pp. 614–645, 2008.

[30] J. W. Messner and P. Pinson, "Online adaptive lasso estimation in vector autoregressive models for high dimensional wind power forecasting," *International Journal of Forecasting*, vol. 35, no. 4, pp. 1485–1498, 2019.

[31] R. P. Monti, C. Anagnostopoulos, and G. Montana, "Adaptive regularization for lasso models in the context of nonstationary data streams," *Statistical Analysis and Data Mining*, vol. 11, no. 5, pp. 237–247, 2018.

[32] Ó. Fontenla-Romero, B. Guijarro-Berdiñas, D. Martinez-Rego, B. Pérez-Sánchez, and D. Peteiro-Barral, "Online machine learning," in *Efficiency and Scalability Methods for Computational Intellect*. IGI Global, 2013, pp. 27–54.

[33] H. M. Gomes, A. Bifet, J. Read, J. P. Barddal, F. Enembreck, B. Pfharinger, G. Holmes, and T. Abdessalem, "Adaptive random forests for evolving data stream classification," *Machine Learning*, vol. 106, no. 9-10, pp. 1469–1495, 2017.

[34] D. A. Bodenham and N. M. Adams, "A comparison of efficient approximations for a weighted sum of chi-squared random variables," *Statistics and Computing*, vol. 26, no. 4, pp. 917–928, 2016.

[35] J. Plasse and N. Adams, "Handling delayed labels in temporally evolving data streams," in *2016 IEEE International Conference on Big Data (Big Data)*, 2016, pp. 2416–2424.

[36] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[37] S. Rosset and J. Zhu, "Piecewise linear regularized solution paths," *The Annals of Statistics*, vol. 35, no. 3, pp. 1012–1030, 2007.

[38] J. Friedman, T. Hastie, H. Höfling, R. Tibshirani *et al.*, "Pathwise coordinate optimization," *The Annals of Applied Statistics*, vol. 1, no. 2, pp. 302–332, 2007.

[39] J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *Journal of Statistical Software*, vol. 33, no. 1, p. 1, 2010.

[40] S.-J. Kim, K. Koh, S. Boyd, and D. Gorinevsky, "ℓ_1 trend filtering," *SIAM review*, vol. 51, no. 2, pp. 339–360, 2009.

[41] M. Buckley and G. Eagleson, "An approximation to the distribution of quadratic forms in normal random variables," *Australian & New Zealand Journal of Statistics*, vol. 30, no. 1, pp. 150–159, 1988.

[42] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[43] S. W. Roberts, "Control chart tests based on geometric moving averages," *Technometrics*, vol. 1, no. 3, pp. 239–250, 1959.

[44] K. M. Carter and W. W. Streilein, "Probabilistic reasoning for streaming anomaly detection," in *2012 IEEE Statistical Signal Processing Workshop (SSP)*. IEEE, 2012, pp. 377–380.

[45] H. Raza, G. Prasad, and Y. Li, "Ewma model based shift-detection methods for detecting covariate shifts in non-stationary environments," *Pattern Recognition*, vol. 48, no. 3, pp. 659–669, 2015.

[46] V. Ishimtsev, A. Bernstein, E. Burnaev, and I. Nazarov, "Conformal k-nn anomaly detector for univariate data streams," *Proceedings of Machine Learning Research*, vol. 60, pp. 1–15, 13–16 Jun 2017.

[47] A. Kejariwal, "Twitter engineering: Introducing practical and robust anomaly detection in a time series [online blog], 2015," *https://blog.twitter.com/engineering/en_us/a/2015/introducing-practical-and-robust-anomaly-detection-in-a-time-series.html*.

[48] B. Rosner, "Percentage Points for a Generalized ESD Many-Outlier Procedure," *Technometrics*, vol. 25, no. 2, pp. 165–172, 1983.

[49] W. S. Cleveland, "Robust locally weighted regression and smoothing scatterplots," *Journal of the American Statistical Association*, vol. 74, no. 368, pp. 829–836, 1979.

[50] A. Y. Kim, C. Marzban, D. B. Percival, and W. Stuetzle, "Using labeled data to evaluate change detectors in a multivariate streaming environment," *Signal Processing*, vol. 89, no. 12, pp. 2529–2536, 2009.

[51] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.

[52] J. Lewis, "Economic impact of cybercrime - no slowing down," *Santa Clara: McAfee & CSI (Center for Strategic and International Studies)*, 2018.

[53] M. J. M. Turcotte, A. D. Kent, and C. Hash, "Unified Host and Network Data Set," *arXiv preprint arXiv:1708.07518*, 2017.

[54] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer Networks*, vol. 51, no. 12, pp. 3448–3470, 2007.

[55] N. Adams and N. Heard, *Data analysis for network cyber-security*. World Scientific Publishing, 2014.

[56] N. A. Heard and N. M. Adams, *Dynamic networks and cyber-security*. Imperial College Press, 2016.

[57] N. Heard, N. Adams, P. Rubin-Delanchy, and M. Turcotte, *Data Science for Cyber-Security*. World Scientific Publishing, 2018.

[58] M. Hebiri, "Sparse conformal predictors," *Statistics and Computing*, vol. 20, no. 2, pp. 253–266, 2010.