IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 31, 2019, accepted August 21, 2019, date of publication September 12, 2019, date of current version October 11, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2940816*

# Real-Time Video Content Popularity Detection Based on Mean Change Point Analysis

**SOTIRIS SKAPERAS[1], (Student Member, IEEE), LEFTERIS MAMATAS[1], (Member, IEEE), AND ARSENIA CHORTI[2], (Member, IEEE)**

[1]Department of Applied Informatics, University of Macedonia, 546 36 Thessaloniki, Greece
[2]ETIS/Université Paris Seine, Université Cergy-Pointoise, ENSEA, CNRS, 95000 Cergy, France

Corresponding author: Lefteris Mamatas (emamatas@uom.edu.gr)

This work was supported in part by the EU's Horizon 2020 Research and Innovation Programme through the 4th Open Call Scheme of the FED4FIRE+ under Grant 732638, in part by the EU-BRA Horizon 2020 NECOS Project under Grant 777067, in part by the Project Measuring Mobile Broadband Networks in Europe (MONROE, H2020), General Secretariat for Research and Technology (GSRT) for the years 2016–2017 (award for the participation in competitive EU projects), under Agreement 644399, and in part by the Ministry of Education, Research, and Religious Affairs, General Secretariat for Research and Technology (GSRT), Greece.

**ABSTRACT** Video content is responsible for more than 70% of the global IP traffic. Consequently, it is important for content delivery infrastructures to rapidly detect and respond to changes in content popularity dynamics. In this paper, we propose the employment of on-line change point (CP) analysis to implement real-time, autonomous and low-complexity video content popularity detection. Our proposal, denoted as *real-time change point detector (RCPD)*, estimates the existence, the number and the direction of changes on the average number of video visits by combining: (i) off-line and on-line CP detection algorithms; (ii) an improved time-series segmentation heuristic for the reliable detection of multiple CPs; and (iii) two algorithms for the identification of the direction of changes. The proposed detector is validated against synthetic data, as well as a large database of real YouTube video visits. It is demonstrated that the RCPD can accurately identify changes in the average content popularity and the direction of change. In particular, the success rate of the RCPD over synthetic data is shown to exceed 94% for medium and large changes in content popularity. Additionally, the dynamic time warping distance, between the actual and the estimated changes, has been found to range between 20 samples on average, over synthetic data, to 52 samples, in real data. The rapid responsiveness of the RCPD is instrumental in the deployment of real-time, lightweight load balancing solutions, as shown in a real example.

**INDEX TERMS** Video content popularity detection, change point analysis, on-line change point detection, binary segmentation algorithm, load balancing.

## I. INTRODUCTION

Video content is projected to account for 82% of the global Internet traffic by 2020, significantly increased from 72% in 2016 [1]. In parallel, novel emerging networking, cloud and edge computing paradigms with significant elasticity capabilities appeared recently, e.g., software-defined networks (SDN) [2], cloud orchestration proposals [3] and content distribution networks (CDNs) [4]. These advances offer the means to respond quickly to changes in content popularity dynamics with appropriate adaptations, e.g., in terms of efficient server resource allocation schemes, load balancing or content caching. As a result, the early detection of changes in content popularity [5], [6] is proving a highly important topic and can have a significant impact on the network traffic and the utilization of servers.

So far, the vast majority of research efforts have focused on the *prediction* of content popularity dynamics, as opposed to their *real time detection*, which is the focus of this study. There is a multitude of reasons as to why the precision of even state-of-the-art prediction algorithms can be impaired. A variety of factors – both from the digital and the physical world – can influence the users' Internet surfing behavior, e.g., [5]: (i) the quality, type (e.g., commercial or user-provided) and life-time of content; (ii) its relevance to users and physical events; (iii) the social interactions between users; and

The associate editor coordinating the review of this manuscript and approving it for publication was Yassine Maleh.

(iv) the content promotion strategies involved. Importantly, mid-term and long-term content popularity prediction [7] – and corresponding adaptations in the network or cloud environment – can prove highly inaccurate [8] and thus result in sub-optimal service planning, provisioning, and utilization of resources or violation of service level agreements.

In this work, to address the aforementioned shortcomings of the commonly employed prediction algorithms, we propose a corresponding detector, referred to as the "real-time change point detector" (RCPD). The RCPD is compatible with modern, flexible networking and cloud approaches, that are highly adaptive and can respond to short-term network dynamics. With accurate, on-line content popularity detection, discrepancies between inaccurate predictions and actual changes can be alleviated. The RCPD is real-time, lightweight, accurate and is parameterized autonomously by analyzing historical data.

In the RCPD, we employ the change point (CP) detection theory and algorithms; their suitability is confirmed against a large number of synthetic as well as real YouTube video datasets. In this contribution, the early detection of changes in the average content popularity is addressed with a novel CP detection methodology, consisting of a training phase, using historical data, and, an on-line phase. In the training phase, we employ a modified off-line CP detection scheme to configure the on-line (sequential) algorithm's parameters. This approach is shown to greatly improve the accuracy of the on-line detector, as in essence, the algorithm parameterization is not arbitrary but rather extracted from corresponding historical data. To the best of our knowledge, it is the first time in the literature that retrospective (off-line) and sequential (on-line) CP detection schemes are combined in a single algorithm operating autonomously (i.e., without manual configuration of parameters).

Besides that, our approach complements the off-line scheme with an improved time-series segmentation heuristic for the detection of multiple CPs. Furthermore, we propose two possible variations for the on-line CP algorithm, the first based on the standard cumulative sum (CUSUM) procedure [9] and the second on the ratio-type CUSUM procedure [10][1]. Additionally, we introduce two alternative indicators to detect the direction of changes: the first one is directly derived from the statistical test of the on-line CP procedure, while the second is based on a modified exponential moving average filter, extensively used in econometrics. As discussed in Sections III and IV, the RCPD combines all the above mentioned algorithmic elements, and is based on sufficiently general and convenient assumptions. Moreover, unlike other approaches e.g., [11], we employ methods that allow dependence between observations (in the form of $t-$dependence), leading to more realistic assumptions for the statistical structure of the content visits.

[1]The advantage of ratio-type CUSUM is that it does not require the estimation of long-run covariance (variance) matrices, which is the case for the standard CUSUM method.

We evaluate the proposed detector and its individual algorithmic components (i.e., the off-line / on-line test statistics, the time-series segmentation algorithm and the trend indicator), over synthetic and real YouTube content views data. Our experiments using synthetic data, generated by an autoregressive moving average (ARMA) filter, demonstrate:

- The superior performance of the proposed time-series segmentation heuristic over the standard approach, improving the true alarm rates by up to 43%.
- The ability of the two proposed trend indicators to identify the direction of estimated changes, with successful identification rates exceeding 99%, in all cases.
- The RCPD performance; the true alarm rates surpass 94% for medium / large changes in the mean number of content views, while the corresponding CP identification lag ranges between 10 to 20 instances, confirming the real-time operation of the detector. On the other hand, the RCPD achieves very small false alarm rates, well within the limits of the statistical error specified by the chosen significance level of the CP algorithms.

Furthermore, our tests on real YouTube content views datasets show that:

- YouTube video views match the underlying assumptions of the RCPD, i.e., the content popularity time-series datasets can be modeled as $t$-dependent.
- The RCPD can detect CPs in more than 70% of the videos in our dataset, implying a sufficiently high number of content popularity changes and the suitability of the CP theory framework for content popularity detection.
- The successful CP direction identifications exceed 91%, i.e., the proposed trend indicators work for real data.
- The average dynamic time warping (DTW) distance [12], [13] between the identified CPs and a benchmark off-line algorithm was estimated to be 52 time instances on average, showcasing the rapid responsiveness of the RCPD.
- The overall processing cost of the RCPD is very low; notably, it took less than one second to process 882 videos on a typical personal computer (PC).

Finally, as a proof-of-concept, we demonstrate the applicability of the proposed algorithm in a real load balancing scenario. We provide a set of measurements showcasing improvements in terms of the clients' connectivity time to download specific content, without a significant impact on the utilization of the content servers. This is achieved due to the deployment of additional content caches, an event triggered by the output of the proposed RCPD detector.

The rest of the paper is organized as follows. In Section II, we discuss our approach with respect to related works. In Section III, we present the training phase of the RCPD algorithm, while the on-line phase is discussed in Section IV. In Section V, we present four experiments over synthetic data, providing an extensive validation of the RCPD and its subroutines, while in Section VI, we discuss corresponding experiments using a database of real YouTube video views.

In Section VII, we demonstrate the load balancing gains achieved through the use of the RCPD, in a realistic content provisioning scenario. Our conclusions and directions for future work are presented in Section VIII.

## II. RELATED WORKS

In this Section, we discuss how this work relates to the literature of video content popularity prediction, on one hand, and, anomaly detection (i.e., CP analysis), on the other hand.

The topic of content popularity attracted a lot of attention in recent years, because of its importance in a number of applications, such as network dimensioning (e.g., capacity planning or scaling of resources), on-line marketing (e.g., advertising, recommendation systems) or real-world outcome prediction (e.g., analysis of economical trends) [5]. The main approaches used for content popularity estimation can be categorized as: (i) cumulative growth studies, estimating the ''amount of attention'' from the publication instance to the prediction moment [6]; (ii) temporal analysis approaches, i.e., how content visits evolve over time [14]; and (iii) clustering methods of content with similar popularity trends [7]. We note that many content popularity studies consider the aggregate behavior of a particular content, e.g., [6], [14], whereas we study the real-time behavior of video views time-series. In addition, studies using clustering methods [7] are based on content popularity prediction and adopt parametric models, unlike the RCPD algorithm that is non-parametric.

To the best of our knowledge, our earlier conference paper [15] is the first in the literature proposing CP techniques [16] for content popularity detection. The RCPD algorithm falls into the general category of anomaly detection [17]; in essence, we assume that no changes in popularity constitutes the normal behavior of video content and search for deviations from this behavior. Non-parametric anomaly detection has typically been considered for the detection of abnormalities in the network traffic. As an example, in [18] an algorithm was proposed based on the Shiryaev-Roberts procedure for anomaly detection in computer network traffic. In [19] and [20], CUSUM based approaches were introduced for the detection of SYN attacks.

Further examples of parametric anomaly detection methods include [21], in which a bivariate sequential generalized likelihood ratio test (LRT) was proposed, accounting for the packet rate – assumed to follow a Poisson distribution – and the packet size – assumed to follow a normal distribution. Other parametric anomaly detection approaches assume a particular underlying process for the normal behavior and search for anomalies on the residuals of the process. For example, in [22], Kalman filtering is combined with several CP methods, such as CUSUM and LRT, to detect anomalies in origin-destination flows. In [23], traffic flows (in the form of TCP's finite state machine), are modeled using Markov chains and an anomaly detection mechanism based on the generalized LRT algorithm is developed.

As opposed to previous content popularity prediction works, in this paper we introduce a novel CP detection methodology that provides accurate, lightweight, autonomous and on-line CP detection of content popularity. We formulate the detection of a change in the average content popularity as a statistical hypothesis test and employ non-parametric procedures to avoid a particular distribution assumption (such as a specific copula model). This context ensures low convergence time since it avoids estimating a large number of model parameters and restrictive assumptions that may not match the structure of the time-series. Furthermore, we avoid problems of parametric models that require parameters' fitting and selection, which become challenging as new data become available. In the proposed RCPD algorithm, an off-line phase specifies important parameters for the on-line phase; these parameters are re-evaluated dynamically after a detected CP. Our load-balancing experiments, elaborated in [4], demonstrate the RCPD's behavior in a real test-bed deployment.

Up to now there are only a handful of proposals addressing the challenges of new flexible networking and cloud architectures accounting for content popularity. Exceptions include [24] in which a logistic-loss machine learning approach to content popularity prediction is applied for a Fog RAN environment, and, our recent papers [4] and [15]. In [4], the algorithm – outlined in [15] and presented extensively in the present – is integrated into an elastic CDN framework based on lightweight cloud capabilities using Unikernels. [4] focuses on the platform details rather than on the CP algorithm; it confirms experimentally the suitability of the latter for relevant flexible network and cloud architectures. The first detailed description of the proposed CP detection algorithm is presented in the following Sections, along with a rich set of validation results. We elaborate on the two phases of the RCPD in Sections III and IV respectively and provide the corresponding pseudo-code.

## III. TRAINING (OFF-LINE) PHASE

In this Section, the training phase of the algorithm is discussed and the fundamental components of the off-line scheme are presented. We note that standard off-line CP schemes can only detect a single CP. To address the issue of detection of multiple CPs, we modify the basic algorithm with a novel time-series segmentation heuristic, that belongs to the family of binary segmentation algorithms.

### A. BASIC OFF-LINE APPROACH

Let $\{X_n : n \in \mathbb{N}\}$ be a sequence of $r$- dimensional random vectors (r.v.). The first dimension represents the number of views for a specific video content within a time period $n \in \{1, \ldots, N\}$, while the other dimensions could be optionally used to represent other content popularity features, such as likes, comments, etc. We assume that $X_1, ..., X_N$ can be written as,

$$X_n = \mu_n + Y_n, \quad 1 \leqslant n \leqslant N \tag{1}$$

where $\{\mu_n : n \in \mathbb{N}\}$ is the mean value of video visits, $\{Y_n : n \in \mathbb{N}\}$ a random component with zero mean $E[Y_n] = 0$ and positive definite covariance matrix, $E\left[Y_n Y_n^T\right] = \Sigma$, while $E[\cdot]$ denotes expectation. We further assume that the time-series is $t$-dependent, implying that for $t_1, t_2, t \in \mathbb{N}$, $Y_{t_1}$ is independent of $Y_{t_2}$ if $|t_1 - t_2| > t$.

The model in (1) and the underlying assumption of $t-$dependence are in agreement with statistical characterizations of the distribution of visits, which have been shown in numerous analyses to follow either a Zipf [25] or a Zipf-Mandelbrot [26] distribution for both commercial and user-generated content. Furthermore, it is confirmed in the real YouTube datasets used in the present work through the evaluation of the time-series's Hurst exponents, as will be discussed in Section VI-A.

The off-line analysis tests the constancy (or not) of the mean values up to the current time $N$. Hence, we define the following null hypothesis of constant mean,

$$H_0 : \mu_1 = \ldots = \mu_N,$$

against the alternative,

$$H_1 : \mu_1 = \ldots = \mu_{k^*_{off}} \neq \mu_{k^*_{off}+1} = \ldots = \mu_N,$$

indicating that the mean value changed at the unknown (time) point $k^*_{off} \in \{1, \ldots, N\}$.

Considering (1) and the corresponding assumptions for the stochastic process $X_n$, we develop a non-parametric CUSUM test statistic following [27]. The test statistic $TS_{off}$, can be viewed as a max-type procedure,

$$TS_{off} = \max_{1 \leqslant n \leqslant N} C_n^T \widehat{\Omega}_N^{-1} C_n, \tag{2}$$

where the parameter $C_n$ is the retrospective CUSUM detector,

$$C_n = \frac{1}{\sqrt{N}} \left( \sum_{i=1}^{n} X_i - n\overline{X}_{1,N} \right), \tag{3}$$

while $\overline{X}_{1,N} = \frac{1}{N}\sum_{i=1}^{N} X_i$ denotes the sample mean. $\widehat{\Omega}_N$ represents a suitable estimator of the long-run covariance $\Omega$, where

$$\Omega = \sum_{i=-\infty}^{\infty} \mathbf{Cov}\left(X_n X_{n-i}\right). \tag{4}$$

The estimator should satisfy,

$$\widehat{\Omega}_N \xrightarrow{P} \Omega \tag{5}$$

where $\xrightarrow{P}$ denotes convergence in probability.

Several estimators have been proposed in the literature that satisfy (5), including kernel-based [28], bootstrap-based [29], etc. Considering our requirement for real-time detection (low computational time), a kernel-based estimator is more suitable; in this context, we employ the Bartlett estimator, so that

$$\widehat{\Omega}_N = \widehat{\Sigma}_0 + \sum_{w=1}^{W} k_{BT}\left(\frac{w}{W+1}\right)\left(\widehat{\Sigma}_w + \widehat{\Sigma}_w^T\right), \tag{6}$$

which satisfies (5), while the function $k_{BT}(.)$ corresponds to the Bartlett weight,

$$k_{BT}(x) = \begin{cases} 1 - |x|, & \text{for } |x| \leqslant 1 \\ 0, & \text{otherwise,} \end{cases} \tag{7}$$

and $\widehat{\Sigma}_w$ denotes the empirical auto-covariance matrix for lag $w$,

$$\widehat{\Sigma}_w = \frac{1}{N} \sum_{n=w+1}^{N} \left(X_n - \overline{X}\right)\left(X_{n-w} - \overline{X}\right)^T. \tag{8}$$

Finally, we chose $W = \log_{10}(N)$ as in [28].

The long-run covariance is involved in the test statistic to incorporate the dependence structure of the r.v. into the statistical analysis, through the integration of second order statistical properties. This approach is suitable for the targeted context since we avoid a restrictive assumption for the dependence structure of the observations.

Going back to the basic question of rejecting or not $H_0$, we need to obtain critical values, denoted by $cv_{off}$, for the test statistic. We approach this issue by considering the asymptotic distribution of the test statistic under $H_0$,

$$TS_{off} \xrightarrow{D} cv_{off} = \sup_{0 \leqslant t \leqslant 1} \sum_{j=1}^{r} B_j^2(t) \quad (N \to \infty), \tag{9}$$

where $\xrightarrow{D}$ denotes convergence in distribution, while $\left(B_j(t) : t \in [0,1]\right), 1 \leqslant j \leqslant r$, are independent standard Brownian bridges $B(t) = W(t) - tW(1)$, and $W(t)$ denotes the standard Brownian motion with mean 0 and variance $t$. The critical values for several significance levels $\alpha$ can be computed using Monte Carlo simulations that approximate the paths of the Brownian bridge on a fine grid. The last step is to estimate the unknown CP, defined previously as $k^*_{off}$, under $H_1$, given by:

$$\hat{k}^*_{off} = \frac{1}{N} \underset{1 \leqslant n \leqslant N}{\operatorname{argmax}} TS_{off}. \tag{10}$$

### B. EXTENDED OFF-LINE APPROACH

The above hypothesis test identifies the existence of at most one CP and does not ensure that the sample remains statistically stationary in either direction of the detection. In particular, by construction (see (2)), the off-line test statistic detects the CP with the highest magnitude. Therefore, for the detection of multiple CPs we need to rephrase the hypothesis test $H_1$, as follows:

$$H_1 : \mu_1 = \ldots = \mu_{k_1} \neq \mu_{k_1+1} = \ldots = \mu_{k_2} \neq \ldots$$
$$\cdots \neq \mu_{k_{\tau-1}+1} = \ldots = \mu_{k_\tau} \neq \mu_{k_\tau+1} = \ldots = \mu_N.$$

A greedy technique to identify multiple CPs is the binary segmentation (BS) algorithm. The standard BS algorithm relies on the general concept of binary segmentation and is an extension of the single CP estimator. First, a single CP is searched for in the time-series. In case of no change, the procedure stops and $H_0$ is accepted. Otherwise, the detected

**Algorithm 1** Modified Binary Segmentation (MBS)

```
procedure MBS(start,end,A)
    ; A: BS method selection (0: standard, 1: modified)
    ; TS_off: the off-line test statistic (eq. 2)
    ; cv_off: the critical value (eq. 9)
    ; k̂*_off: the identified CP (eq. 10)
    calculate TS_off(start, end) and cv_off
    if TS_off(start, end) > cv_off then
        calculate k̂*_off and store it in array s
        MBS(start,k̂*_off,0)
        MBS(k̂*_off+1,end,0)
    end if
    if array_length(s) > 0 and A=1 then
        Ŝ ← {1} ∪ {s} ∪ {N} ; N: the time-series length
        for i=2:N-1 do
            MBS(Ŝ_{i-1}, Ŝ_{i+1},0)
            keep in l the validated CPs only
        end for
    end if
end procedure
```

CP is used to divide the time-series into two segments in which new searches are performed. The procedure is iterated until no more CPs are detected. The BS algorithm is lightweight (computational time $O(N\log N)$), while its conceptual simplicity leads to efficient implementations. On the other hand, it has been shown in the literature [30], [31], that the standard BS algorithm tends to overestimate the number of CPs, as it does not cross-validate them after their detection.

In the extended off-line approach, we propose the modification of the standard BS with a cross-validation step of the estimated CPs. The cross-validation step is similar to that used in the iterative cumulative sum of squares (ICSS) segmentation algorithm [32], which is used to search for CPs on the marginal variance of independent and identically distributed (i.i.d.) r.vs. In the extended off-line algorithm we consider the CPs estimated from the standard BS in pairs and check if $H_0$ is rejected in the segment delimited by each pair. If $H_0$ is not rejected in a particular segment, then no change can be detected in it; as a result, all CPs that fall in the respective segment are eliminated. The improvement, in terms of accuracy, is shown through simulation results in Section IV. The pseudo-code of the modified BS algorithm is given in *Algorithm 1*; note that we integrate the algorithm with the test statistic $TS_{off}$, given in equation (2) and the corresponding critical value ($cv_{off}$) given in (9).

## IV. ON-LINE PHASE

In this Section, we describe the on-line scheme that includes: (i) two alternative CUSUM-type approaches for the detection of a change in the mean; and (ii) two alternative approaches to estimate the direction of a change.

### A. ON-LINE ANALYSIS

We rewrite equation (1) in the form,

$$X_n = \begin{cases} \mu + Y_n, & n = 1, \ldots, m + k^* - 1 \\ \mu + Y_n + I, & n = m + k^*, \ldots \end{cases} \tag{11}$$

where $\mu, I \in \mathbb{R}^r$ represents the mean parameters before and after the unknown time of possible change $k^* \in \mathbb{N}^*$ respectively. As a reminder, the first dimension of the time-series represents the video views; the rest could be likes, comments, etc., and $\{Y_n : n \in \mathbb{N}\}$ is a random component. The term $m \in \mathbb{N}$ denotes the length of the training period, i.e., an interval of length $m$ over the historical period during which the mean is assumed to remain unchanged, so that,

$$\mu_1 = \cdots = \mu_m. \tag{12}$$

To satisfy this assumption, the modified off-line CP test previously presented is run in order to identify a suitable $m$. With $m$ determined, the on-line procedure can be used to check whether (12) holds as new data become available.

In the form of a statistical hypothesis test, the on-line problem becomes,

$$\begin{aligned} H_0 &: I = 0, \\ H_1 &: I \neq 0. \end{aligned} \tag{13}$$

The on-line sequential analysis belongs to the category of stopping time stochastic processes. In general, a chosen on-line test statistic $TS_{on}(m, l)$ and a given threshold $F(m, l)$ define the stopping time $\tau(m)$:

$$\tau(m) = \begin{cases} \min\{l \in \mathbb{N} : TS_{on}(m, l) \geqslant F(m, l)\}, \\ \infty, \text{ if } TS_{on}(m, l) < F(m, l)\ \forall l \in \mathbb{N}, \end{cases} \tag{14}$$

implying that $TS_{on}(m, l)$ is calculated on-line for every $l$ in the monitoring period. The procedure stops if the test statistic exceeds the value of the threshold function $F(m, l)$. As soon as this happens, the null hypothesis is rejected and a CP is detected. The following properties should hold for $\tau(m)$,

$$\lim_{m\to\infty} Pr\{\tau(m) < \infty | H_0\} = \alpha,$$

ensuring that the probability of false alarm is asymptotically bounded by $\alpha \in (0, 1)$, and,

$$\lim_{m\to\infty} Pr\{\tau(m) < \infty | H_1\} = 1,$$

ensuring that under $H_1$ the asymptotic power of the statistical test is unity. The threshold $F(m, l)$ is given by,

$$F(m, l) = cv_{on,a} g(m, l), \tag{15}$$

where: (i) the critical value $cv_{on,a}$ is determined from the asymptotic behavior of the stopping time procedure under $H_0$ by letting $m \to \infty$; and (ii) the weight function,

$$g(m, l) = \sqrt{m}\left(1 + \frac{l}{m}\right)\left(\frac{l}{l+m}\right)^{\gamma} \tag{16}$$

depends on the sensitivity parameter $\gamma \in [0, 1/2)$.

We use two different CUSUM approaches; the standard [9], with test statistic denoted by $TS^{ct}_{on}$, and, the ratio-type [10], with test statistic denoted by $TS^{rt}_{on}$. Their corresponding critical values are denoted by $cv^{ct}_{on,a}$ and $cv^{rt}_{on,a}$, respectively, and their stopping rules by $\tau_{ct}(m)$ and $\tau_{rt}(m)$, correspondingly. Both tests are based on the sequential CUSUM detector, $E(m, l)$,

$$E(m,l) = \left(\overline{X}_{m+1,m+l} - \overline{X}_{1,m}\right) \tag{17}$$

The standard CUSUM test is expressed as:

$$TS^{ct}_{on}(m,l) = l\widehat{\Omega}_m^{-\frac{1}{2}} E(m,l), \tag{18}$$

where $\widehat{\Omega}_m$ is the estimated long-run covariance, defined as in (4), that captures the dependence between observations. Then, the stopping rule $\tau_{ct}(m)$, is defined as:

$$\tau_{ct}(m) = \min\{l \in \mathbb{N} : \|TS^{ct}_{on}(m,l)\|_1 \geq cv^{ct}_{on,a} g(m,l)\}, \tag{19}$$

where the $\ell_1$ norm is involved to modify $TS^{ct}_{on}$ so that it can be compared to a one dimensional threshold function. The critical value, $cv^{ct}_{on,a}$, is derived from the asymptotic behavior of the stopping rule under $H_0$:

$$\begin{aligned}
&\lim_{m\to\infty} Pr\{\tau(m) < \infty\} \\
&\quad = \lim_{m\to\infty} Pr\left\{ \sup_{1\leqslant l\leqslant\infty} \frac{\|TS^{ct}_{on}(m,l)\|_1}{g(m,l)} > cv^{ct}_{on,\alpha} \right\} \\
&\quad = Pr\left\{ \sup_{t\in[0,1]} \frac{\|W(t)\|_1}{t^\gamma} > cv^{ct}_{on,\alpha} \right\} = \alpha.
\end{aligned} \tag{20}$$

Unlike standard CUSUM tests, ratio type statistics do not require to estimate the long-run covariance and are also considered for this reason in this analysis. The precise form of the chosen statistic is given in the following quadratic form,

$$\begin{aligned}
&TS^{rt}_{on}(m,l) \\
&= \frac{l^2}{m} E^T(m,l) \left\{ \frac{1}{m^2} \sum_{j=1}^{m} j^2 \left(\overline{X}_{1,j} - \overline{X}_{1,m}\right)\left(\overline{X}_{1,j} - \overline{X}_{1,m}\right)^T \right\}^{-1} \\
&\quad \times E(m,l),
\end{aligned} \tag{21}$$

with its equivalent stopping rule,

$$\tau_{rt}(m) = \min\{l \in \mathbb{N} : TS^{rt}_{on} \geq cv^{rt}_{on,a} g^2(m,l)\}. \tag{22}$$

Similarly to the standard CUSUM, the critical value, $cv^{rt}_{on,a}$, is estimated by,

$$\begin{aligned}
&\lim_{m\to\infty} Pr\{\tau(m) < \infty\} \\
&\quad = Pr\left\{ \sup_{t\in[0,\infty)} \Delta_\gamma(t) > cv^{rt}_{on,\alpha} \right\} \\
&\quad = \alpha,
\end{aligned} \tag{23}$$

where,

$$\begin{aligned}
\Delta_\gamma(t) &= \frac{1}{\eta^2_\gamma(t)} B^T(1+t) \left( \int_0^1 B(r) B^T(r) dr \right)^{-1} B(1+t), \\
\eta^2_\gamma(t) &= (1+t)\left(\frac{t}{1+t}\right)^\gamma,
\end{aligned}$$

and $B(t)$ is a standard Brownian bridge, $t \in [0, \infty)$ .

Similarly to the off-line case, the on-line critical values for both test statistics can be computed using Monte Carlo simulations, considering that,

$$cv^{ct}_{on,\alpha} = \sup_{t\in[0,1]} \frac{W(t)}{t^\gamma}, \tag{24}$$

$$cv^{rt}_{on,\alpha} = \sup_{t\in[0,\infty)} \Delta_\gamma(t). \tag{25}$$

The estimated on-line CP, $\hat{k}^*_{on}$, is derived directly from the value of the stopping time $\tau(m)$, as,

$$\hat{k}^*_{on} = m + \{\tau(m) | \tau(m) < \infty\}. \tag{26}$$

### B. TREND INDICATOR

Considering the on-line procedure, the hypothesis $H_1$ in (13) is two-tailed because the test statistics $TS^{rt}_{on}$ and $TS^{ct}_{on}$ are formulated in a quadratic form and a $\ell_1$ norm, respectively. This means that the stopping time rule $\tau_{ct}(m)$ (or $\tau_{rt}(m)$) cannot be an indicator of the direction of a detected change. Thus, to estimate the direction of a change we introduce two indicators: i) based on the CUSUM detector in (17), denoted by $TI_{ts}$; and ii) based on the moving average convergence divergence (MACD) filter [33], denoted by $TI_f$.

Focusing on $TI_{ts}$, the indicator is directly derived from the form of the sequential CUSUM detector $E(m, l)$. As shown in (17), the detector compares the mean value of the observations that are collected on-line for a chosen monitoring period $l$, with the mean value of a subsample of the historical data over the predetermined training sample. Hence, for a detected CP, we have that,

$$\begin{cases} E(m,l) > 0, & \text{denotes an upward change} \\ E(m,l) < 0, & \text{denotes a downward change.} \end{cases} \tag{27}$$

However, in certain cases, limiting the window over which the direction of a change is estimated to the immediate neighbourhood of a detected CP can be unreliable due to the continuous variability of the time-series. In such cases, we have to estimate the direction of a change by incorporating more elaborate filters; in this context, we estimate the direction of detected changes by applying the MACD indicator. The MACD is based on an exponential moving average (EMA) filter, of the form,

$$EMA_p(n) = \frac{2}{p+1} X_n + \frac{p-1}{p+1} EMA_p(n-1), \tag{28}$$

with $p$ denoting the lag parameter. The MACD series can be derived from the subtraction from a short $p_2$ lag EMA (sensitive filter) of a longer $p_3$ lag EMA (blunt filter), as described below:

$$MACD(n) = EMA_{p_2} - EMA_{p_3}. \tag{29}$$

The trend indicator $TI_f$ is then obtained by the subtraction of a short $p_1$ lag EMA filter of a MACD series from the raw MACD series, as described below:

$$TI_f(n) = MACD(n) - EMA_{p_1}(MACD(n)), \quad p_1 < p_2 < p_3. \tag{30}$$

In the evaluation of $TI_f$ three exponential filters are involved. In essence, $TI_f$ is an estimation of the second derivative over an interval around the change (considering that the subtraction of a filtered variable from the variable generates an estimate of its time derivative). In contrast to other works [33], we only adopt $TI_f$ to characterize the direction from the specific value of $TI_f$ at the estimated time of change. We announce an upward change if $TI_f(\hat{k}^*_{on}) > 0$, otherwise, if $TI_f(\hat{k}^*_{on}) < 0$, a downward change.

Finally, we propose a modification of the trend indicator $TI_f$, converting it from a point estimator to an interval estimator; instead of evaluating $TI_f(\hat{k}^*_{on})$, we propose to evaluate the trend indicator at a time interval $(\hat{k}^*_{on}, \hat{k}^*_{on}+h)$, where $h$ is a threshold parameter:

$$TI_f(\hat{k}^*_{on}, h) = \sum_{l=\hat{k}^*_{on}}^{\hat{k}^*_{on}+h} TI_f(l). \tag{31}$$

The proposed $TI_f(\hat{k}^*_{on}, h)$ modification improves the estimator's accuracy; the calculation of the sum of a multitude of observations, after a CP, can smooth out a potential false one-point estimation, especially in the case of small changes.

### C. OVERALL ALGORITHM

We outline in *Algorithm 2* the RCPD algorithm, as a combination of the off-line and the on-line phase, in the form of pseudo-code. Beginning from the initial value set for the monitoring starting period, denoted by $m_s$, the modified off-line algorithm is applied over the whole historical period; the training period $m$ is then defined as the interval elapsed from the last detected off-line CP (if one exists) to $m_s$. In pseudo-code this step is described in lines $14-18$. As a second step, the on-line test statistic, $TS_{on}(m, l)$ in (14), is applied for a specified monitoring time frame $l$. If a content popularity change is detected at time instance $\hat{k}^*_{on}$, the trend indicator subroutine is called to reveal the direction of change.[2] At this point the procedure stops and a new starting point for the monitoring window is defined as $m_s = \hat{k}^*_{on} + d$, where $d$ is a constant value specifying a period assuming no change. This step is described in lines $19-29$. Otherwise, if no change is detected after a maximum of $l$ instances, the procedure restarts from the last time point, $m_s = m_s + l$.

**Algorithm 2** The Real-time CP Detector (RCPD)

```
1:  procedure RCPD(X_n,m_s,k)
2:      ; X_n: time-series of video views
3:      ; m_s: running end of training period
4:      ; m: training period
5:      ; l: monitoring time frame
6:      ; d: period assuming no change
7:      ; TS_on: on-line test statistic (eq. 18 or 21)
8:      ; cv_on: critical value (eq. 24 or 25)
9:      ; k̂*_on: the estimated on-line CP (eq. 26)
10:     ; TI: trend indicator (TI_ts or TI_f)
11:     for n in X_n do
12:         if n = m_s then
13:             s = MBS(1,m_s,1) ; calculate off-line CPs
14:             if array_length(s) > 0 then
15:                 m = {max(s), m_s} ; max(s) is the latest CP
16:             else
17:                 m = {max(1, m_s − u), m_s} ; u a large
    value
18:             end if
19:         else if m_s < n < m_s + l then
20:             calculate TS_on(m, l)
21:             if TS_on(m, l) > cv_on then
22:                 calculate TI
23:                 signal CP and estimated direction
24:                 m_s = ĉp_on + d ; keep a distance from ĉp_on
25:             end if
26:         else if n = m_s + l then
27:             m_s = m_s + l ; start a new training period
28:         end if
29:     end for
30: end procedure
```

## V. VALIDATION OF THE RCPD USING SYNTHETIC DATA

In this Section, we validate the performance of the overall algorithm by performing a series of four different experiments on synthetic data. The use of synthetic data allows us to regulate the parameters of the time-series in terms of mean changes and thus obtain quantitative metrics for the performance of the proposed algorithms.

The choice of the time-series model for the generation of the synthetic data is based on the fact that several studies have shown that ARMA models capture very well content popularity evolution. For example, in [7] it has been concluded that an ARMA model can efficiently describe the daily access patterns of YouTube content, based on an extensive analysis of $100,000$ videos. Similarly, in [34] an ARMA model has been proposed for the estimation of the popularity of video content. Motivated by these findings, for the validation of the proposed algorithm we use an ARMA$(1, 1)$ time-series. We generate $1,000$ time-series of length $N = 600$ samples. Without loss of generality, we assume an initial mean value $\mu_0 = 0$, noting that the performance of the RCPD is independent of the initial mean value and only depends on the magnitude of the variation of the mean value before and after a CP.

In the first experiment, we begin with a comparison of the standard BS to the proposed modified BS algorithms described in Section II-B. We perform two tests; in the first test we introduce two CPs at the instances $k^*_i = (iN)/3$, $i = 1, 2$, while in second test, we introduce four CPs at $k^*_i = (iN)/5, i = 1, \ldots, 4$. The two tests are repeated for three different values of the magnitude of a change $\mu_1 = 1$,

[2] In the load balancing scenario discussed in Section VII, in the case of an increase in the content popularity a new content cache is being deployed, while conversely a decrease leads to the removal of an existing cache.

**TABLE 1. Percentage of the successful CP detections for the standard and modified BS algorithm.**

| | Test 1: two CPs | | Test 2: four CPs | |
|---|---|---|---|---|
| $\mu$ | BS | modified BS | BS | modified BS |
| | True (false) alarm rate | | True (false) alarm rate | |
| $\mu_1$=1 | 0.94 (0.06) | 0.95 (0.05) | 0.5 (0.258) | 0.7 (0.05) |
| $\mu_2$=1.5 | 0.95 (0.05) | 0.95 (0.05) | 0.5 (0.258) | 0.9 (0.08) |
| $\mu_3$=2 | 0.95 (0.05) | 0.95 (0.05) | 0.47 (0.53) | 0.9 (0.1) |

**TABLE 2. Success rates of trend indicators.**

| | Test 1: two CPs | | Test 2: four CPs | |
|---|---|---|---|---|
| $\mu$ | $TI_{ts}$ | $TI_f$ | $TI_{ts}$ | $TI_f$ |
| | Success rate | | Success rate | |
| $\mu_1$=1 | 0.99 | 0.99 | 0.99 | 0.99 |
| $\mu_2$=1.5 | 1 | 1 | 1 | 1 |
| $\mu_3$=2 | 1 | 1 | 1 | 1 |

$\mu_2 = 1.5, \mu_3 = 2$, i.e., we randomly increase or decrease the mean value by $\mu_j, j = 1, \ldots, 3$ at the time of change. Table 1 summarizes our findings regarding the true and false alarm rates of the two algorithms.

Both the standard and the modified BS algorithms provide similar true alarm rates, exceeding 94%, in the first test. On the contrary, in the more challenging second test, the superiority of the modified BS over the standard BS algorithm is clear. The modified BS algorithm achieves true alarm rates in excess of 70%, even in the demanding scenario of a relatively small change in the mean $\mu_1 = 1$. On the other hand, the standard BS algorithm has in all cases a true alarm rate of less than 50%, rendering any CP detection highly questionable. The second test confirms that the standard BS algorithm is prone to an overestimation of the number of CPs as shown by the high false alarm rates (in excess of 25% in all cases), an issue that can be effectively addressed by the modified BS algorithm which scores false alarm rates below 10%.

Next, in the second experiment, using the same test sets as above, we measure the success rates achieved by the proposed trend indicators $TI_{ts}$ in (27) and $TI_f$ in (31) for $h = 0$ (larger thresholds provided the same true identification rates). The results are summarized in Table 2. The two trend indicators successfully identify the direction of a change in more than 99% of the cases, which shows that they can be interchangeably employed. In the assessment of the performance using real datasets in Section VI, we solely employ the $TI_f$ trend indicator.

We proceed by assessing the proposed RCPD algorithm using both the standard and the ratio type CUSUM. In this third experiment, we measure the average number of CPs detected, averaged over 1, 000 simulations when a single CP is introduced in the ARMA time-series at the time instance $\frac{N}{2} = 300$. We consider different values for the magnitude of change $\mu \in \{0, 0.5, 0.7, 1, 1.2, 1.5, 2\}$ and the monitoring window length $l \in \{25, 50, 100\}$. We note that we included the case $\mu = 0$ – which corresponds to the absence of a change – to evaluate the false alarm rate of the overall algorithm. We omit results with true alarm rates lower than 50% as they are statistically unreliable. In terms of the remaining algorithmic parameters, we have set the minimum distance between two successive CPs to $d = 50$,[3] the sensitivity parameter to $\gamma = 0.25$ [35] (we choose a neutral value as the behaviour of $\gamma$ is well studied), and, the significance level to $\alpha = 0.05$. In each test of the third experiment we measure the exact number of CPs detected, tabulated as one the following three values: i) 0 when (falsely[4]) no CP is detected; ii) 1 when (correctly) a single CP is detected; and iii) $> 1$ when (falsely) multiple CPs are detected. Finally, we measure the median of the time instance of the single CP detection, denoted by $\hat{k}^*$.[5] The results of this experiment are presented in Table 3 and are discussed below.

Firstly, we observe that both the standard and the ratio type CUSUM achieve very small false alarm rates, inferior to 6% when no CP is inserted, irrespective of the choice of $l$. On the contrary, the choice of $l$ readily affects the algorithm's success rate for $\mu > 0$; for small changes in the mean value, $\mu = 0.5, 0.7$, a larger monitoring window $l$ increases the algorithm's true alarm rates in identifying correctly the existence of the CP. For medium and high changes in the magnitude of change $\mu = 1, 1.2, 1.5, 2$, it is observed that a high true alarm rate – in excess of 93% for the standard CUSUM – is achieved, while choosing a smaller $l$ can slightly increase the true alarm rates. As a result, depending on the application, a choice of a larger $l$ can be appropriate if the algorithm is to be employed as a universal CP detector. Alternatively, a smaller $l$ can be chosen when the focus is on the identification of large changes in the mean value, i.e., we are interested primarily in detecting CPs of larger magnitude.

Secondly, we observe that overall, the ratio type CUSUM is outperformed by the standard CUSUM in all tests. Consequently, the standard CUSUM based detector can be considered as an efficient universal choice. Finally, we observe that the lag between $\hat{k}^*$ and the actual instance of change at the point 300 decreases with increasing $\mu$, ranging from 343 to 307, while it appears less sensitive to changes in $l$. This demonstrates that, intuitively, larger magnitude changes can be detected faster. This result is important for load balancing applications as it provides us with the means to quickly respond to significant changes in the network traffic.

Subsequently, in Table 4 in the following page, we present the outputs of the fourth experiment in which we assess the performance, averaged over 1, 000 simulations, of the RCPD algorithm when two CPs are inserted in the ARMA

[3]This choice is justified by our observations of the minimum distance between successive CPs in real data sets, presented in Section VI.

[4]Except for the $\mu = 0$ case.

[5]We omit the results with true detection rate lower than 50%.

**TABLE 3.** Results of the RCPDs' algorithm CPs detection for one change in the mean value.

| | | ARMA(1,1) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $\mu$ | $l$ | standard CUSUM | | | | ratio-type CUSUM | | | |
| | | Number of detected CPs | | | $\hat{k}^*$ | Number of detected CPs | | | $\hat{k}^*$ |
| | | 0 | 1 | $> 1$ | med | 0 | 1 | $> 1$ | med |
| $\mu = 0$ | 25 | 0.95 | 0.05 | 0 | - | 0.95 | 0.05 | 0 | - |
| | 50 | 0.95 | 0.05 | 0 | - | 0.95 | 0.05 | 0 | - |
| | 100 | 0.94 | 0.06 | 0 | - | 0.95 | 0.05 | 0 | - |
| $\mu = 0.5$ | 25 | 0.7 | 0.29 | 0.01 | - | 0.8 | 0.19 | 0.01 | - |
| | 50 | 0.16 | 0.8 | 0.04 | 343 | 0.55 | 0.43 | 0.02 | - |
| | 100 | 0 | 0.93 | 0.07 | 341 | 0.2 | 0.76 | 0.04 | 348 |
| $\mu = 0.7$ | 25 | 0.26 | 0.73 | 0.01 | 332 | 0.69 | 0.3 | 0.01 | - |
| | 50 | 0 | 0.96 | 0.04 | 326 | 0.3 | 0.65 | 0.05 | 328 |
| | 100 | 0.01 | 0.91 | 0.08 | 331 | 0.05 | 0.89 | 0.06 | 335 |
| $\mu = 1$ | 25 | 0.01 | 0.97 | 0.02 | 327 | 0.52 | 0.46 | 0.02 | - |
| | 50 | 0 | 0.96 | 0.04 | 316 | 0.08 | 0.86 | 0.06 | 321 |
| | 100 | 0 | 0.92 | 0.08 | 321 | 0 | 0.95 | 0.05 | 323 |
| $\mu = 1.2$ | 25 | 0.01 | 0.97 | 0.02 | 323 | 0.43 | 0.54 | 0.03 | 331 |
| | 50 | 0 | 0.95 | 0.05 | 316 | 0.02 | 0.93 | 0.05 | 317 |
| | 100 | 0 | 0.93 | 0.07 | 318 | 0 | 0.93 | 0.07 | 318 |
| $\mu = 1.5$ | 25 | 0 | 0.97 | 0.03 | 320 | 0.36 | 0.6 | 0.04 | 329 |
| | 50 | 0 | 0.95 | 0.05 | 310 | 0 | 0.94 | 0.06 | 313 |
| | 100 | 0 | 0.93 | 0.07 | 314 | 0 | 0.94 | 0.06 | 318 |
| $\mu = 2$ | 25 | 0 | 0.97 | 0.03 | 310 | 0.26 | 0.71 | 0.03 | 317 |
| | 50 | 0 | 0.95 | 0.05 | 307 | 0 | 0.93 | 0.07 | 310 |
| | 100 | 0 | 0.94 | 0.06 | 310 | 0 | 0.94 | 0.06 | 313 |

time-series. We introduce a change at the time instance $k_1^* = \frac{N}{3} = 200$ and a second CP at the time instance $k_2^* = \frac{2N}{3} = 400$. We investigate the true and false alarm rates for $\mu \in \{0.5, 0.7, 1, 1.2, 1.5, 2\}$ and $l \in \{25, 50, 100\}$, while the rest of the parameters retain the values of the third experiment. In each test of the fourth experiment we measure the exact number of CPs detected, tabulated as one the following three values: i) $< 2$ when (falsely) less than two CPs are detected, ii) 2 when (correctly) two CPs are detected, and iii) $> 2$ when (falsely) more than two CPs are detected. Finally, we measure the median of the detection instances of the two CPs, denoted by $\hat{k}_1^*$ and $\hat{k}_2^*$, respectively (we omit the results with true detection rate lower than 50%).

Similarly to the third experiment, we observe that increasing $l$ increases the true alarm rates for small magnitudes in the mean changes $\mu = 0.5, 0.7$, while this trend is reversed in high magnitudes $\mu = 1.5, 2$. For medium values $\mu = 1, 1.2$ the effect of $l$ on the true alarm rates is less than 2%. Furthermore, in agreement with the outputs of the third experiment, with increasing $\mu$ the algorithms achieve increasingly high success rates, over 93% for the standard CUSUM when $\mu \geq 1$.

In addition, the superior performance of the standard CUSUM is re-confirmed in all the tests of the fourth experiment. Finally, with respect to the lag in the estimation of the time instances of the CPs, we observe that, as in experiment three, larger magnitude changes can be detected faster, e.g., for $\mu = 2$ a lag inferior to 11 instances is observed for both CPs with the standard CUSUM, irrespective of $l$.

Concluding this Section, we have presented an extensive set of experiments that provide strong evidence for the efficiency of the proposed algorithms. We have explicitly demonstrated the superiority of the modified BS over the standard BS algorithm and confirmed the validity of the proposed trend indicators. Subsequently, we evaluated the performance of the overall algorithm for various values of $\mu$ and $l$. We have shown that the RCPD algorithm achieves extremely high true alarm rates for larger values of $\mu$, while increasing the length of the monitoring window $l$ can significantly impact the performance for small values of $\mu$. Finally, overall, the standard type CUSUM outperforms the ratio type CUSUM and should be preferred.

## VI. PERFORMANCE EVALUATION USING REAL DATA

In this Section we investigate the performance of the proposed algorithms using a real dataset provided within the framework of the CONGAS project [36]; the dataset consists of the number of views of 882 YouTube videos, observed over $N = 1,000$ instances.

**TABLE 4.** Results of the RCPDs' algorithm CPs detection for two mean changes.

| | | ARMA(1,1) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\mu$ | $l$ | standard CUSUM | | | | | ratio-type CUSUM | | | | | |
| | | Number of detected CPs | | | $\hat{k}_1^*$ | $\hat{k}_2^*$ | Number of detected CPs | | | $\hat{k}_1^*$ | $\hat{k}_2^*$ | |
| | | $< 2$ | 2 | $> 2$ | med | | $< 2$ | 2 | $> 2$ | med | | |
| $\mu_1 = 0.5$ | 25 | 0.88 | 0.12 | 0 | - | - | 0.95 | 0.05 | 0 | - | - | |
| | 50 | 0.38 | 0.60 | 0.02 | 251 | 440 | 0.79 | 0.2 | 0.01 | - | - | |
| | 100 | 0.1 | 0.87 | 0.03 | 242 | 443 | 0.54 | 0.44 | 0.02 | - | - | |
| $\mu_1 = 0.7$ | 25 | 0.41 | 0.58 | 0.01 | 230 | 427 | 0.9 | 0.1 | 0 | - | - | |
| | 50 | 0.06 | 0.91 | 0.03 | 223 | 427 | 0.58 | 0.41 | 0.01 | - | - | |
| | 100 | 0.01 | 0.93 | 0.06 | 227 | 428 | 0.25 | 0.72 | 0.03 | 231 | 439 | |
| $\mu_1 = 1$ | 25 | 0.04 | 0.93 | 0.03 | 219 | 420 | 0.74 | 0.25 | 0.01 | - | - | |
| | 50 | 0.03 | 0.93 | 0.04 | 215 | 419 | 0.26 | 0.71 | 0.03 | 221 | 423 | |
| | 100 | 0 | 0.94 | 0.06 | 217 | 420 | 0.05 | 0.9 | 0.05 | 220 | 424 | |
| $\mu_1 = 1.2$ | 25 | 0.01 | 0.96 | 0.03 | 214 | 414 | 0.56 | 0.42 | 0.02 | - | - | |
| | 50 | 0 | 0.95 | 0.05 | 212 | 416 | 0.17 | 0.79 | 0.04 | 215 | 428 | |
| | 100 | 0 | 0.94 | 0.06 | 217 | 420 | 0.02 | 0.93 | 0.05 | 216 | 421 | |
| $\mu_1 = 1.5$ | 25 | 0 | 0.98 | 0.02 | 211 | 411 | 0.33 | 0.63 | 0.04 | 213 | 417 | |
| | 50 | 0 | 0.94 | 0.06 | 209 | 413 | 0.1 | 0.85 | 0.05 | 213 | 415 | |
| | 100 | 0 | 0.94 | 0.06 | 211 | 415 | 0 | 0.96 | 0.04 | 216 | 419 | |
| $\mu_1 = 2$ | 25 | 0 | 0.98 | 0.02 | 208 | 407 | 0.12 | 0.85 | 0.03 | 210 | 412 | |
| | 50 | 0 | 0.95 | 0.05 | 207 | 410 | 0.3 | 0.91 | 0.06 | 209 | 413 | |
| | 100 | 0 | 0.94 | 0.06 | 209 | 411 | 0 | 0.96 | 0.04 | 211 | 414 | |

### A. STATISTICAL PROPERTIES OF THE REAL DATASET

First, we evaluate the validity of the most important underlying assumption of this analysis, that the content popularity can be modelled as the sum of a constant mean and a weak-dependent ($t$-dependent) stochastic process, as given in (1). A first intuitive method to test whether the time-series is short-range dependent (SRD) is through its autocorrelation function (ACF). The ACF for a weakly-stationary process $\{X_t : t \in \mathbb{N}\}$ with mean value $\mu$ is given by,

$$\rho(k) = \frac{(X_t - \mu)(X_{t+k} - \mu)}{\sigma^2}.$$

Note that if $\sum_{k=-\infty}^{\infty} \rho(k) \rightarrow \infty$ the process has long-range dependence (LRD), while if $\sum_{k=-\infty}^{\infty} |\rho(k)| < \infty$ it exhibits SRD. To distinguish between these two phenomena, we use the following functional form of the ACF,

$$\rho(k) \sim C_i^{2H-2}, \quad \text{as } i \rightarrow \infty,$$

where $C_i > 0$ and $H \in (0, 1)$ is the Hurst exponent characterizing the LRD,i.e., $H \in (1/2, 1)$ indicates the presence of LRD. It is challenging to accurately estimate the Hurst exponent out of real data [37] and several methods have been proposed in the literature [38]. In this work, we apply two semi-parametric tests, identified as accurate options among others presented in the survey paper [38]. The first method uses the discrete second order derivative in the time domain while the second uses the discrete second order derivative in the wavelet domain. Both methods estimate an $H \leq 0.5$ for 95% of the YouTube time-series, indicating the validity of our assumptions related to the equation (1).

### B. PERFORMANCE OF THE OFF-LINE TRAINING PHASE

First, we test the hypothesis $H_0$ of no change in the mean structure on our dataset. $H_0$ is rejected in approximately 70% of the cases, for a significance level of $a = 0.05$. This outcome indicates that CP algorithms can identify changing content dynamics in real times series.

Next, we estimate the number of CPs, by applying the extended off-line algorithm. The corresponding results are illustrated in Fig. 1 and indicate a sufficiently high number of content popularity anomalies (i.e., mean changes). Hence, a CP analysis is indeed a suitable tool for content popularity detection.

To evaluate the performance of the proposed trend indicator $TI_f$, we need a baseline independent assessment of the direction of change. We declare that a real increase in the mean value of content visit exists if

$$\mathsf{E}[X(\hat{k}^*_{i-1,off}) : X(\hat{k}^*_{i,off})] < \mathsf{E}[X(\hat{k}^*_{i,off}) : X(\hat{k}^*_{i+1,off})], \quad (32)$$

or, that a real decrease in the number of visits exists if

$$\mathsf{E}[X(\hat{k}^*_{i-1,off}) : X(\hat{k}^*_{i,off})] > \mathsf{E}[X(\hat{k}^*_{i,off}) : X(\hat{k}^*_{i+1,off})], \quad (33)$$

where $i = 1, \cdots, card(\hat{k}^*_{off})$, $\hat{k}^*_0 = 1$, $\hat{k}^*_{s+1} = N$ and $\mathsf{E}[\cdot]$ denotes the numerical average. We test the modified MACD $TI_f$ on two sets of videos. The first set, Video

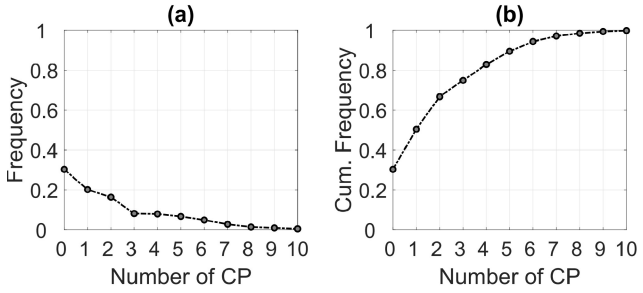



**FIGURE 1.** Estimated a) frequency and b) cumulative frequency of the number of CPs per time-series.

**TABLE 5.** Success rates of $TI_f$ trend indicator.

| h | 0 | 3 | 5 | 7 | 10 |
|---|---|---|---|---|---|
| Video Set 1 | 0.69 | 0.91 | 0.95 | 0.97 | 0.98 |
| Video Set 2 | 0.90 | 0.99 | 0.99 | 0.99 | 0.99 |

Set 1, comprises the whole dataset, while the second set, Video Set 2, comprises only the videos with a considerable average number of visits ($>$ 10), i.e., for which, $\mathsf{E}[X(1) : X(1000)] > 10$.

The percentage of successful $TI_f$ identifications are tabulated in Table 5 for five values of the parameter $h$, namely $h = 0,\ 3,\ 5,\ 7$ and 10, where $h$ denotes the $TI_f$'s calculation threshold introduced in Section IV-B. Commenting on the results for Video Set 1, the $TI_f$ trend indicator works well, except for $h = 0$, providing at least 90% correct direction identifications. As expected, as $h$ increases the procedure works better. More specifically, an $h \geq 5$ parameter choice yields a success rate of 95%, while if a more agile estimation is needed then an $h \geq 3$ still maintains a 91% accuracy. Considering the interim time between consecutive changes, we deduce that an $h \leq 7$ is preferable. Regarding Video Set 2, we see that the results are highly improved, indicating that the procedure works even better for the most popular videos. In practice, this represents the more interesting scenario as it will have a greater impact in terms of the applied load balancing mechanism.

Furthermore, in Fig. 2, the time instances of upward and downward changes are shown in the form of a boxplot. It is intuitive that upward changes occur earlier than downward changes. Moreover, Fig. 2 demonstrates that the multitude of upward changes is greater than the respective of downward changes, indicating that decreases in popularity are sharper than increases. In particular, we estimated that out of the total number of changes, 67% are upward.

Finally, we analyze the interim time between consecutive CPs. The results presented in Fig. 3 illustrate the existence of a sufficiently large gap between consecutive potential changes. 90% of the intervals corresponding to consecutive CPs exceed 70 time instances and only 5% of them are shorter than 50 time instances, ensuring that a sufficiently

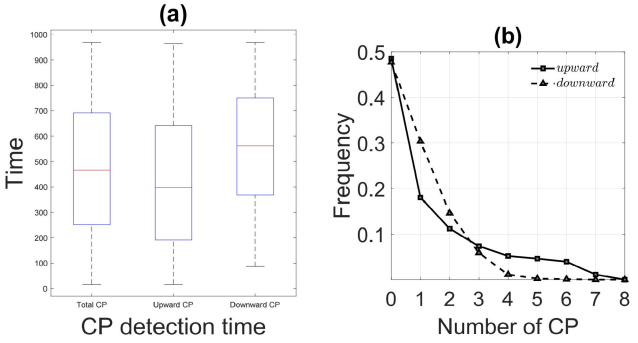



**FIGURE 2.** Frequency values of the number of upward and downward CPs, per time-series.

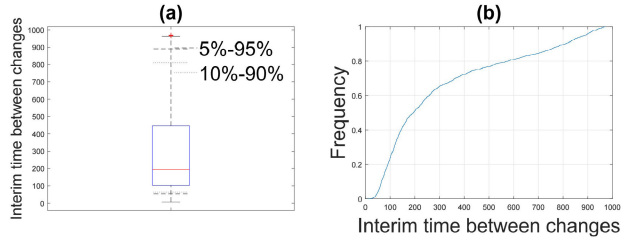



**FIGURE 3.** a) Boxplot including the interval (5% – 95%) (dashed line) and (10% – 90%) interval (dotted line), b) Cumulative frequency for the interim time of consecutive CPs.

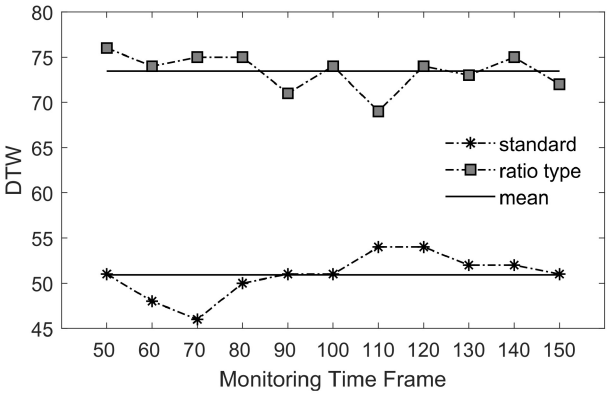



**FIGURE 4.** DTW distances for the two on-line detection schemes.

large training window can be applied. The results depicted in Fig. 3 allow adjusting parameters of the on-line phase, in particular the minimum time interval between consecutive changes, denoted by the parameter $d$.

### C. EVALUATION OF THE RCPD ALGORITHM

In the previous subsection we have evaluated the performance of the off-line algorithm and demonstrated its efficiency as well as how it is employed in determining parameters of the on-line phase, such as the interval assuming no change $d$ and the threshold parameter of $TI_f$ $h$.

We further employ the off-line algorithm as a benchmark against which the performance of the RCPD algorithm will be evaluated. We note that the off-line analysis provides the *best possible statistical detection* of the actual mean changes,

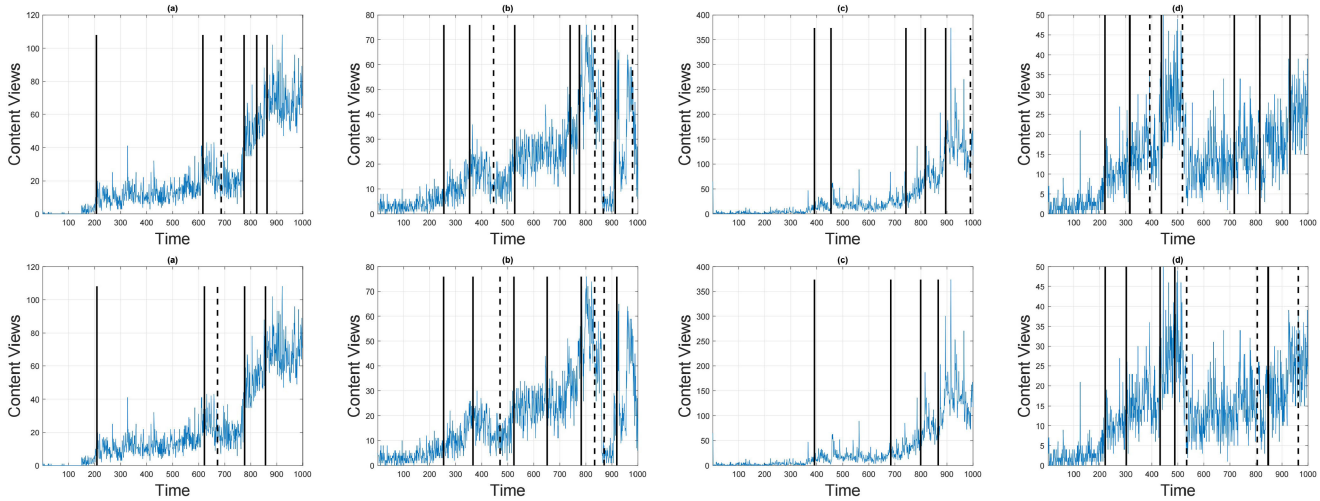



**FIGURE 5.** Outputs of the RCPD algorithm; using standard CUSUM (upper row) and ratio type CUSUM (lower row) for four different time-series. Solid and dashed lines depict an upward and a downward change, respectively.

as off-line algorithms operate retrospectively over the entirety of each of the time-series. Thus, in absence of a priori knowledge of the actual CPs in the real data (as opposed to the synthetic data in which the CPs were controlled), we evaluate the performance of the RCPD procedure by measuring the ''similarity'' of its outputs (detected CPs, instances of detection and trends) to the corresponding outputs of the off-line version.

As the number of detected CPs and / or their exact positions are likely to differ at the output of the retrospective (off-line) and of the RCPD algorithm, in order to obtain a measure of their similarity, we estimate their dynamic time warping (DTW) distance. The DTW is a dynamic programming tool that measures distances between asynchronous sequences and is widely used by the speech processing community [12].

The results are presented in Fig. 4, where the estimated DTW distances are depicted for several values of the monitoring window length $l \in [40, 150]$, to investigate the consistency of parameter $l$ over different values. In the RCPD algorithm we use $d = 50$ (minimum distance between two changes) and have set the sensitivity parameter to $\gamma = 0.25$. The estimated mean DTW distance for the standard CUSUM is 52 and for the ratio-type CUSUM is 73. For comparison purposes, we note that the corresponding DTW distance over the synthetic data is 20 for medium / large changes, while the true CP detections are around 95%. As a result, we can infer, that the outputs of the on-line algorithm, using the standard CUSUM, are ''very close'' to the outputs of the benchmark off-line algorithm. In agreement with our observations over the synthetic data, the DTW distance using the ratio-type CUSUM is clearly larger.

We also study the magnitude of the detected CPs. We define as the CP magnitude the percentage-wise change in the mean values before and after the CP. We group the measured magnitudes for all change points using the four percentile threshold values 10%, 15%, 25% and 50%, i.e., reflecting the frequency of magnitudes exceeding the respective thresholds. The results are summarized in Table 6. According to our results, both the standard and ratio type CUSUM algorithms detect the most significant changes in the content popularity. Moreover, ratio-type CUSUM detects, in general, CPs with the largest magnitude of change, in agreement with synthetic data results.

**TABLE 6.** Empirical percentiles of mean values change rate.

| | Percentiles Threshold | | | |
|---|---|---|---|---|
| | 10% | 15% | 25% | 50% |
| Standard | 9% | 13.1% | 20.8% | 42.21% |
| Ratio type | 9.5% | 14.82% | 28.22% | 67.40% |

Additionally, for illustration purposes, we depict the RCPD algorithm's outputs for four different time-series. We set the beginning of the monitoring period at $m_s = 200$ and monitoring horizon $l = 50$, the on-line parameter $g = 0.25$ and the significance level to $a = 0.05$. The corresponding results are depicted in Fig. 5, showing the estimated CPs by applying the standard CUSUM and the ratio type CUSUM procedures, respectively. In both cases, the estimated changes correspond to the real content popularity changes; visual inspection suggests that the performance of the standard CUSUM is more reasonable (e.g., Fig. 5d). The RCPD, as it is illustrated in Fig. 5b seems to be adaptable to ''fast'' changes; without getting ''confused'' by random peaks in the time-series, such as those in Fig. 5a or in Fig. 5c.

### D. TIME DEPENDENCIES OF PIECEWISE TIME-SERIES

We also measure the autocorrelation function of the piecewise - divided by the detected CPs - time-series. Results are

**TABLE 7. Percentages of time-series with time dependencies exceeding $t$ samples.**

| $t$ | $\geq 1$ | $\geq 5$ | $\geq 15$ | $\geq 30$ | $\geq 50$ |
|---|---|---|---|---|---|
| piecewise | 0.93 | 0.57 | 0.23 | 0.05 | 0.04 |

tabulated in Table 7 and verify the short dependence structure of the dataset; significant lags in time dependencies higher than 30 instances can be found in less than 5% of the time-series. Furthermore, the fact that the ACF of the piecewise time-series drops to zero quickly indicates that the detected CPs split the time-series into stationary segments, which, additionally, confirms indirectly the accuracy of the off-line CP estimations over the changes in the real data.

### E. COMPUTATIONAL COMPLEXITY AND SCALABILITY

Finally, we present a MATLAB ® implementation of the overall algorithm with a large number of time-series (882 in this experiment) to quantify its performance in terms of processing cost. The computational time is measured on a Lenovo IdeaPad 510-15IKB laptop, with an Intel Core i7-7500U @ 2.70 GHz processor and 12 GB RAM. In Fig. 6, we show the aggregate processing cost per time instance for the two on-line methods and the total number of time-series. For the first 100 time instances, the algorithm collects the initial data, since it bootstraps. The peaks indicate the off-line part of the algorithm, which is more processing demanding mainly due to the segmentation algorithms running in parallel. The on-line part in the standard on-line algorithm indicates a linear complexity, since it is based on (18), while the equivalent quantity in (21) of the ratio-type is more CPU intensive, justifying the comparatively higher processing cost of the latter algorithm. In both cases, the aggregate processing cost is typically much less than a second, which demonstrates the lightweight nature of the proposed scheme. Such results could be further improved with a distributed deployment of scheme replicas since each of the time-series could be processed independently.

## VII. THE RCPD ALGORITHM IN A LOAD BALANCING SCENARIO

In this Section, we demonstrate our proposal in a real content distribution scenario, balancing the traffic between web clients and content caches with a bespoke DNS-based load-balancer. We implement the RCPD algorithm as a client-server MATLAB ® application. The RCPD engine receives periodic content popularity measurements; if a CP is detected, the corresponding upward or downward changes are signalled to the load balancer. The load balancer: (i) distributes the load between the deployed content caches, in a round-robin fashion; (ii) tracks content visits and communicates them to the RCPD engine; and (iii) deploys or removes content caches based on the RCPD outputs.

We implement the web clients using with the httpperf tool (https://github.com/httperf/httperf). The number of clients at

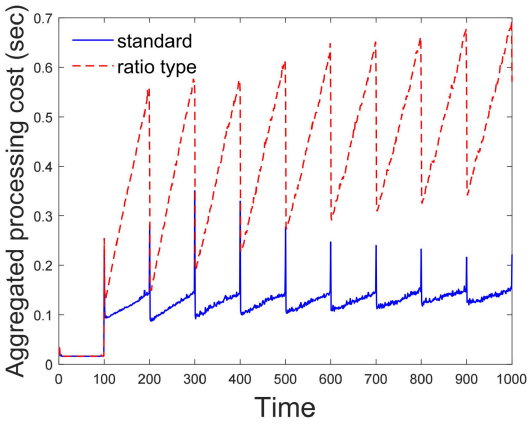



**FIGURE 6. The aggregated overall processing cost, per time-instance, of the RCPD algorithm over 882 time-series.**

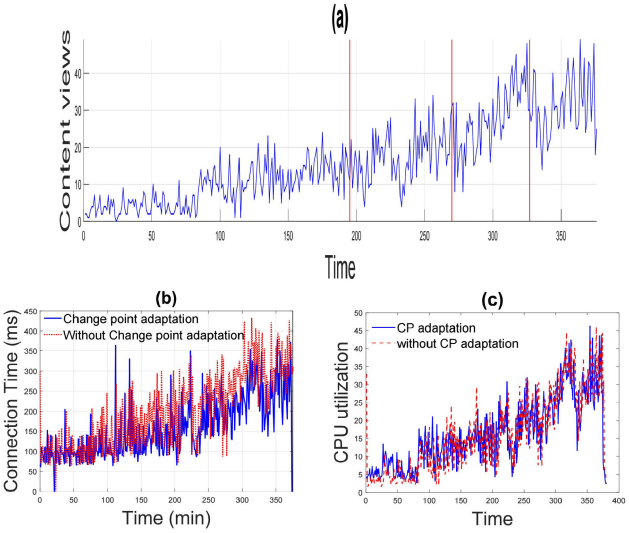



**FIGURE 7. a) time-series of video content views, red lines depict the detected CPs, b) the connection time with and without RCPD adaptation and c) the equivalent servers' CPU utilization.**

each time instance is based on a real time-series of YouTube content views, illustrated in Fig. 7a. In practice, an experimental run without the RCPD mechanisms uses three content caches constantly and a run with the RCPD mechanism enabled uses initially two and then three, four and five content caches, after each of the three detected change points, respectively. As we show in Fig. 7b, the web clients improve their connectivity times to download the content, while as demonstrated in Fig. 7c the CPU utilization in the servers hosting the content remains almost the same. A relevant experimental platform is presented in [4].

## VIII. CONCLUSION AND FUTURE WORK

In this paper, we developed the RCPD, a novel algorithm for the real-time detection of changes in the mean value of content popularity. Approaching the problem statistically, we efficiently combined off-line and on-line non-parametric CUSUM procedures to avoid restrictive assumptions for

content popularity behavior and to reduce the overall computational cost. We divided the algorithm in two phases. The first phase is an extended retrospective (off-line) procedure with a modified BS algorithm and is used to adjust on-line parameters, based on historical data of the particular video. The second phase integrates one of two alternative trend indicators to the sequential (on-line) procedure, to reveal the direction of a detected change. We provided extensive simulations, using synthetic and real data, that demonstrated the performance of the proposed algorithm for the successful identification of content popularity changes in real-time. We also demonstrated through experimental measurements that the RCPD's processing cost is almost imperceptible. Finally we provided proof-of-concept by applying the algorithm in a load balancing application, highlighting its efficiency in a realistic setting.

In future work, we will evaluate the proposed scheme using multi-dimensional time-series to capture more accurately the dynamics of content popularity better (e.g., incorporate additional dimensions with the number of likes, comments, etc.) and in different contexts, such as on the real-time resource utilization of servers. We will also investigate and further extend the algorithm's scalability properties, theoretically and experimentally, i.e., estimate the number of videos that can be analyzed in parallel. Our aspiration is to conduct real large-scale CDN experiments utilizing a distributed architecture with multiple content popularity analyzers, monitoring in real-time clusters of videos at a minimum overall processing cost.

## REFERENCES

[1] G. Zhu, G. Cheng, and K. Wang, "Big data analytics for program popularity prediction in broadcast TV industries," *IEEE Access*, vol. 5, pp. 24593–24601, 2017.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Apr. 2008.

[3] *Necos Project: Towards Lightweight Slicing of Cloud Federated Infrastructures*. Accessed: Mar. 10, 1989. [Online]. Available: http://www.h2020-necos.eu/

[4] P. Valsamas, S. Skaperas, and L. Mamatas, "Elastic content distribution based on unikernels and change-point analysis," in *Proc. 24th Eur. Wireless Conf. (EW)*, Catania, Italy, May 2018, pp. 1–7.

[5] A. Tatar, M. D. de Amorim, S. Fdida, and P. Antoniadis, "A survey on predicting the popularity of Web content," *J. Internet Services Appl.*, vol. 5, no. 1, p. 8, 2014.

[6] G. Szabo and B. A. Huberman, "Predicting the popularity of online content," *Commun. ACM*, vol. 53, no. 8, pp. 80–88, 2010.

[7] G. Gürsun, M. Crovella, and I. Matta, "Describing and forecasting video access patterns," in *Proc. IEEE Int. Conf. Comput. Commun. (IEEE INFOCOM)*, Shanghai, China, Apr. 2011, pp. 16–20.

[8] J. Cheng, L. Adamic, P. A. Dow, J. M. Kleinberg, and J. Leskovec, "Can cascades be predicted?" in *Proc. 23rd Int. Conf. World Wide Web (WWW)*, Seoul, South Korea, Apr. 2014, pp. 925–936

[9] S. Fremdt, "Asymptotic distribution of the delay time in page's sequential procedure," *J. Statist. Planning Inference*, vol. 145, pp. 74–91, Feb. 2014.

[10] Y. Hoga, "Monitoring multivariate time series," *J. Multivariate Anal.*, vol. 155, pp. 105–121, Mar. 2017.

[11] E. Brodsky and B. S. Darkhovsky, *Nonparametric Methods in Change Point Problems*. Dordrecht, The Netherlands: Kluwer, 2013.

[12] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *Proc. AAAI Workshop Knowl. Disc. Databases (KDD)*, Seattle, WA, USA, Aug. 1994, vol. 10, no. 16, pp. 359–370.

[13] R. J. Kate, "Using dynamic time warping distances as features for improved time series classification," *Data Mining Knowl. Discovery*, vol. 30, no. 2, pp. 283–312, Mar. 2016.

[14] H. Pinto, J. M. Almeida, and M. A. Gonçalves, "Using early view patterns to predict the popularity of youtube videos," in *Proc. 6th ACM Int. Conf. Web Search Data Mining (WSDM)*, Rome, Italy, Feb. 2013, pp. 365–374.

[15] S. Skaperas, L. Mamatas, and A. Chorti, "Early video content popularity detection with change point analysis," in *Proc. IEEE Global Commun. Conf. (IEEE GLOBECOM)*, Abu Dhabi, United Arab Emirates, Dec. 2018, pp. 1–7.

[16] M. Basseville and I. V. Nikiforov, *Detection of Abrupt Changes: Theory and Application*, vol. 104. Englewood Cliffs, NJ, USA: Prentice-Hall, 1993.

[17] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, Jul. 2009.

[18] A. G. Tartakovsky, A. S. Polunchenko, and G. Sokolov, "Efficient computer network anomaly detection by changepoint detection methods," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 1, pp. 4–11, Feb. 2013.

[19] A. G. Tartakovsky, B. L. Rozovskii, R. B. Blazek, and H. Kim, "A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods," *IEEE Trans. Signal Process.*, vol. 54, no. 9, pp. 3372–3382, Sep. 2006.

[20] H. Wang, D. Zhang, and K. G. Shin, "Change-point monitoring for the detection of DoS attacks," *IEEE Trans. Depend. Sec. Comput.*, vol. 1, no. 4, pp. 193–208, Oct. 2004.

[21] G. Thatte, U. Mitra, and J. Heidemann, "Parametric methods for anomaly detection in aggregate traffic," *IEEE/ACM Trans. Netw.*, vol. 19, no. 2, pp. 512–525, Apr. 2011.

[22] A. Soule, K. Salamatian, and N. Taft, "Combining filtering and statistical methods for anomaly detection," in *Proc. 5th ACM SIGCOMM Conf. Internet Meas.*, New York, NY, USA, Oct. 2005, p. 31.

[23] I. Nevat, D. M. Divakaran, S. G. Nagarajan, P. Zhang, L. Su, L. L. Ko, and V. L. L. Thing, "Anomaly detection and attribution in networks with temporally correlated traffic," *IEEE/ACM Trans. Netw.*, vol. 26, no. 1, pp. 131–144, Feb. 2018.

[24] Y. Jiang, M. Ma, M. Bennis, F. Zheng, and X. You, "A novel caching policy with content popularity prediction and user preference learning in fog-RAN," in *Proc. IEEE Global Commun. Conf. (GC Wkshps)*, Dec. 2017, pp. 1–6.

[25] X. Zhou and C.-Z. Xu, "Optimal video replication and placement on a cluster of video-on-demand servers," in *Proc. Int. Conf. Parallel Process. (ICPP)*, Vancouver, BC, Canada, Aug. 2002, pp. 547–555.

[26] W. Tang, Y. Fu, L. Cherkasova, and A. Vahdat, "Modeling and generating realistic streaming media server workloads," *Comput. Netw.*, vol. 51, no. 1, pp. 336–356, Jan. 2007.

[27] A. Aue and L. Horváth, "Structural breaks in time series," *J. Time Ser. Anal.*, vol. 34, no. 1, pp. 1–16, Jan. 2013.

[28] D. W. K. Andrews, "Heteroskedasticity and autocorrelation consistent covariance matrix estimation," *Econometrica*, vol. 59, pp. 817–858, May 1991.

[29] D. Wied, "A nonparametric test for a constant correlation matrix," *Econ. Rev.*, vol. 36, no. 10, pp. 1157–1172, Apr. 2017.

[30] M. Lavielle and G. Teyssière, "Adaptive detection of multiple change-points in asset price volatility," in *Long Memory in Economics*, G. Teyssiere and A. Kirkman, Eds. Berlin, Germany: Springer, 2007, pp. 129–156.

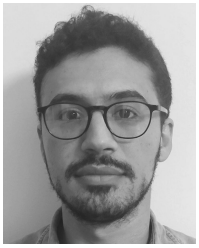
[31] D. Angelosante and G. B. Giannakis, "Sparse graphical modeling of piecewise-stationary time series," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process (IEEE ICASSP)*, Prague, Czech Republic, May 2011, pp. 1960–1963.

[32] C. Inclán and G. C. Tiao, "Use of cumulative sums of squares for retrospective detection of changes of variance," *J. Amer. Stat. Assoc.*, vol. 89, pp. 913–923, Feb. 1994.

[33] H. Kai, Q. Zhengwei, and L. Bo, "Network anomaly detection based on statistical approach and time series analysis," in *Proc. Int. Conf. Adv. Inform. Netw. Appl. Workshops (WAINA)*, Bradford, U.K., May 2009, pp. 205–211.

[34] N. Ben Hassine, R. Milocco, and P. Minet, "ARMA based popularity prediction for caching in content delivery networks," in *Proc. IEEE Wireless Days*, Porto, Portugal, Mar. 2017, pp. 113–120.

[35] D. Wied and P. Galeano, "Monitoring correlation change in a sequence of random variables," *J Statist. Planning Inference*, vol. 143, no. 1, pp. 186–196, Jan. 2013.
[36] M. Zeni, D. Miorandi, and F. De Pellegrini, "YOUStatAnalyzer: A tool for analysing the dynamics of youtube content popularity," in *Proc. 7th Int. Conf. Perform. Eval. Methodol. Tools*, Torino, Italy, Dec. 2013, pp. 286–289.
[37] R. G. Clegg, "A practical guide to measuring the hurst parameter," *Int. J. Simul. Syst. Sci. Technol.*, vol. 7, no. 2, pp. 3–14, Oct. 2006.
[38] J.-M. Bardet, G. Lang, G. Oppenheim, A. Philippe, S. Stoev, M. Taqqu, P. Doukhan, G. Oppenheim, and M. S. Taqqu, "Semi-parametric estimation of the long-range dependence parameter: A survey," in *Theory and Applications of Long-Range Dependence*. Boston, MA, USA: Birkhäuser, 2003, pp. 557–577.



**SOTIRIS SKAPERAS** (S'18) received the B.Sc. degree in mathematics and the M.Sc. degree in statistics and modeling from the Department of Mathematics, Aristotle University of Thessaloniki, Greece, in 2013 and 2016, respectively. He is currently pursuing the Ph.D. degree in the area of resource management in 5G networks from the Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece. He involved in the areas of resource allocation/load balancing for 5G networks using time-series/change point analysis and stochastic modeling.

**LEFTERIS MAMATAS** (S'04–M'08) received the Diploma and Ph.D. degrees from the Department of Electrical and Computer Engineering, Democritus University of Thrace, Greece, in 2003 and 2008, respectively. He is currently an Assistant Professor with the Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece, where he leads the Softwarized & Wireless Networks Research Group. He was a Researcher with the University College London, U.K., the Space Internetworking Center/Democritus University of Thrace, Greece, and the DoCoMo Euro-Labs, Munich, Germany. He has published more than 60 articles in international journals and conferences. His research interests include the areas of software-defined networks, the Internet of Things, 5G networks, and multi-access edge computing. He participated in many international research projects, such as NECOS (H2020), FED4FIRE+ OC4 (H2020), WiSHFUL OC2 (H2020), MONROE OC2 (H2020), Dolfin (FP7), UniverSELF (FP7), and Extending Internet into Space (ESA). He served as the General Chair for the WWIC2016 Conference and the INFOCOM SWFAN 2016 Workshop and as the TPC Chair for the INFOCOM SWFAN 2017, E-DTN 2009, and IFIP WWIC 2012 conferences/workshops. He is a Guest Editor for the *Elsevier Ad Hoc Networks Journal*.

**ARSENIA CHORTI** (S'00–M'05) received the M.Eng. degree in electrical and electronic engineering from the University of Patras, Greece, the D.E.A. degree in electronics from the University Pierre et Marie Curie, Paris VI, France, and the Ph.D. degree in electrical engineering from Imperial College London, U.K., in November 2005. She undertook postdoctoral positions at the University of Southampton, U.K., Technical University of Crete, Greece, and University College London, U.K., from 2005 to 2008. She served as a Senior Lecturer in communications for Middlesex University, U.K., from December 2008 to April 2010. From 2010 to 2013, she was a Marie Curie IOF Researcher with Princeton University, NJ, USA, and with the Institute of Computer Science-FORTH, Greece. From 2013 to 2017, she was a Lecturer with the University of Essex, U.K. She is currently an Associate Professor with ETIS, UMR 8051, University Paris Seine, University Cergy-Pontoise, ENSEA, CNRS, Cergy, France. She is also a Visiting Research Fellow with the University of Essex, U.K. Her work has so far been disseminated in more than 60 journals and international conferences and one book.