Group anomaly detection in mobile app usages: A spatiotemporal convex hull methodology

Agathe Blaise, Mathieu Bouet, Vania Conan, Stefano Secci

Group anomaly detection in mobile app usages: a spatiotemporal convex hull methodology

Agathe Blaise[a], Mathieu Bouet[a], Vania Conan[a], Stefano Secci[b]

[a] *Thales, 92230 Gennevillers, France*
[b] *Cnam, Cedric, 75003 Paris, France*

Abstract

Analysing mobile apps communications can unleash significant information on both the communication infrastructure state and the operations of mobile computing services. A wide variety of events can engender unusual mobile communication patterns possibly interesting for pervasive computing applications, e.g., in smart cities. For instance, local events, national events, and network outages can produce spatiotemporal load anomalies that could be taken into consideration by both mobile applications and infrastructure providers, especially with the emergence of edge computing frameworks where the two environments merge. Being able to detect and timely treat these anomalies is therefore a desirable feature for next-generation cellular and edge computing networks, with regards to security, network and application performance, and public safety. We focus on the detection of mobile access spatiotemporal anomalies by decomposing, aggregating and grouping cellular data usage features time series. We propose a methodology to detect first raw anomalies, and group them in a spatiotemporal convex hull, further refining the anomaly detection logic, with a novel algorithmic framework. We show how with the proposed framework we can unveil details about broad-category mobile events timeline, their spatiotemporal spreading, and their impacted apps. We apply our technique to extensive real-world data and open source our code. By linkage with ground-truth special events that happened in the observed period, we show how our methodology is able to detect them. We also evidence the existence of five main categories of anomalies, finely characterising them. Finally, we identify global patterns in the anomalies and assess their level of unpredictability, based on the nature of the impacted mobile applications.

1. Introduction

Through mobile apps, a telecom operator can witness a wide variety of anomaly events happening, ranging from local (e.g., football matches or concerts) to national events (e.g., political happenings), passing through network outages at both network access and Internet/cloud levels, app updates and failovers. Often, the detection of such events can allow reconfiguring the network to work around the negative effect a given event can have in terms of infrastructure overload and mobile application[1] interruption. Understanding the spatiotemporal characteristics of mobile app usages anomaly and related impacted apps can therefore be useful for both the mobile network operator and application providers. The penetration of multi-access edge computing (MEC) infrastructures, into the access networks, further exacerbates the impact that an in-depth knowledge of mobile app-related anomalies can grant in terms of resource scaling and cloud stack orchestration.

First, the network operator can take into account the spatiotemporal dynamics of mobile access loads during a specific or special event in the way the network and computing infrastructure is operated (e.g., during a match there is more traffic around the stadium, while before and after it, the supporters coming from and going back home being split in different neighbourhoods). Knowing the typology of impacted mobile apps related to a given category of events can support efficient resource allocation, in runtime or even in advance for known events. For instance, it can be useful to allow more bandwidth for streaming apps during bank holidays and lock-downs. The operator can also distinguish between known and unknown events, and in the latter case, thanks to mobile app data analysis, try to understand the root causes and take countermeasures.

Second, understanding per-app spatiotemporal traffic peaks can serve existing and forthcoming pervasive computing applications [8, 29]. For example, navigation systems could avoid known congested areas, and social apps could lead people to or away from the crowds. Other apps include emergency plans, accident or crime zone prediction, and the inference of points of interest. The characterisation of neighbourhoods based on mobile app usages could also allow for real-time maps exploitable for the advertisement or construction industries.

In this paper, we propose a group anomaly detection methodology named Anomaly SpatioTEmporal Convex Hull

Email addresses: agathe.blaise@thalesgroup.com (Agathe Blaise), mathieu.bouet@thalesgroup.com (Mathieu Bouet), vania.conan@thalesgroup.com (Vania Conan), stefano.secci@cnam.fr (Stefano Secci)

[1] referred to as 'mobile app' as well

(*ASTECH*). We define a 3-step algorithm that isolates and groups raw singleton (per: app, cell, timeslot, feature) anomalies as follows:

- It decomposes the features time series, extracts the residual component, and detects anomalies as usage drops and peaks.
- Anomalies are thus grouped into snapshots, which represent the network state at a given place and time. Upon spatiotemporal categorisation of anomalies, it groups the abnormal snapshots both spatially and temporally to get spatiotemporal convex hulls that we call *group anomalies*. This enables to qualify them, unveiling details about their timeline, spatiotemporal spreading, and impacted apps.
- It applies a further clustering to the set of detected group anomalies, to partition them into low-grain categories.

We apply it to extensive real-world data collected by a major mobile network operator in the whole Ile-de-France over three months. We show how our algorithm is able to detect known and unknown anomalies impacting the mobile traffic data during these three months. In particular, leveraging on additional ground-truth event information, we evidence the existence of five categories of anomalies: *local events*, *national events*, *outages*, *bank holiday*, and *app updates*. We also found specific typologies of impacted apps for each category inducing a traffic increase. For the sake of reproducibility, our source code is publicly available at [18]. Our contributions can be summarised as follows.

i) We propose a novel methodology to detect group anomalies in mobile apps usage data and classify them via per-app profiles. Focusing on group anomalies, i.e., spatiotemporal events, differs from detecting traditional anomalies and allows getting insights about their duration, their spatial spreading, and the group of impacted mobile apps. Contrary to previous approaches, the information about app profiles enables us to classify events in addition to detecting them. We then investigate several main questions corresponding to the two research questions below.

ii) Can we identify and classify events happening in the city from the analysis of mobile network use? We show that we are able to identify and group the anomalies that impacted mobile traffic in a given area during a given period. Such anomalies include local events, national events, app failovers or updates, Internet outages, and bank holidays. We rely on three main characteristics to represent an event: the variation of traffic (i.e., more or less traffic than usual), its spatial spreading, and the list of impacted apps during the event. The clustering of the detected events shows that 75% of the events were correctly classified compared to ground-truth labels.

iii) Which correlations exist between the mobile apps that are impacted and the type of detected events? Hereafter, we attempt to model the coarse patterns that we can detect in terms of impacted mobile applications, as well as the slight variations in these models. Our results show that we can easily identify service fail-overs and application updates as they are impacted by a single app. For local and national events, specific typologies of anomalous apps were observed depending on the event category; for example, messaging and streaming apps were usually impacted during matches. We showed that there actually exists typologies of impacted apps depending on the type of events; however slight differences among each category can also be observed. Spatially, variations in intensity occur between central places and suburbs, while temporally, slightly different behaviours can be observed between commuting and working hours.

Our contributions differ from existing special events detection approaches in three main aspects. First, contrary to previous approaches, we are able to finely characterise the identified events, in addition to just detecting them; five classes of special events have been observed with specific characteristics and can be assigned using clustering algorithms. Second, some works in the literature paid attention to the app usage, but never for the purpose of special events detection; we show that the apps impacted during a given event are strongly correlated to the type of event, with implications in city management, since this knowledge can be critical information on which to make decisions about events management and the prediction of services usage. Finally, our algorithm has a polynomial time complexity, which is less than other approaches from the state-of-the-art, especially compared to deep learning-based approaches such as LSTM networks [32] or Gaussian Mixture Models (GMM) [11].

The paper is organised as follows. Sect. 2 presents the related work and positions our work with respect to the state of the art. Sect. 3 introduces the dataset that we employed in our study. In Sect. 4, we describe our methodology. In Sect. 5 we focus on the extraction of raw anomalies, while Sect. 6 presents how we extract and refine group anomalies. In Sect. 7, we present results from the numerical evaluation. In Sect. 8, we qualify the space and time complexity of *ASTECH* and discuss its scalability. Finally, Sect. 9 concludes the paper.

2. Related work

In this section, we review the related work and position our work with respect to the state of the art.

2.1. Detection of spatiotemporal anomalies

The survey [27] reviews large-scale mobile traffic analysis with respect to social, mobility, and network aspects. From a social perspective, demographic, economical, or environmental factors do influence the way users consume mobile apps indeed. The possible relationships between the environment, in terms of both geographical and temporal features, and the communication structure are also

Table 1: Overview of special events detection and comparison to the proposed approach.

Ref	Events			Technique			Granularity			Complexity	Classif
	Social	National	Emergency	Threshold	DL	AD	Place	Feature	App		
[3]	✗	✗	Bombings, outages, natural disasters	✓	✗	✗	BS	Call volume, event duration	✗	Linear	✗
[8]	Sports, arts	✗	✗	✓	✗	✗	Grid space	Users mobility	✗	Linear	✓
[10]	✓	✗	✗	✓	✗	✗	Grid space	Call volume	✗	Linear	✗
[28]	Sports	Bank holidays, political events	✗	✓	✗	✗	✗	✗	✗	Linear	✓
[13]	✓	✗	✗	✗	✗	Clustering	Grid space	Call volume	✗	Polynomial	✗
[25]	Sports, arts	✗	✗	✓	✗	✗	BS	Call volume	✗	Polynomial	✗
[32]	Sports, religious events	✗	✗	✗	LSTM, Autoencoders	✗	BS	Traffic volume	✗	Polynomial	✗
[21]	✗	✗	Cell outage	✗	✗	Semi-supervised statistical-based AD	Grid space	Traffic volume	✗	Polynomial	✗
Our work	Sports, arts	Political events, bank holidays, app update	Outage, fire, bombing	✗	✗	Spatio-temporal group AD	BS	Traffic volume, sparsity, app typology	✓	Polynomial	✓

described. Among them, the authors focus on the *detection of special events*, ranging from political happenings (e.g., elections or manifestations) to entertainment occasions (e.g., concerts, sports games), and accidents (e.g., power outages or exception road congestion).

Authors in [8, 10] propose threshold-based algorithms to detect special events. In [25], the authors analyse human mobility and the resulting dynamics in the network workload caused by different types of large-scale events. They use heat maps to visually analyse the spatiotemporal dynamics of the movement patterns of the participants of the large-scale event, but do not propose a broader model to characterise such events and detect similar ones. Authors in [21] focus on the detection of unusually low user activity to detect cell outages and unusually high user traffic areas to detect the need for additional resource allocation. In [13], authors use the information on residual communication to determine how different geographical areas are affected by the same unusual event. Using a time series decomposition, they exploit the seasonal components to segment the city into clusters with similar patterns, and compare the residual components of similar areas to detect local events. Other approaches [32] use deep learning algorithms to detect spatiotemporal anomalies in mobile traffic data, such as Long Short-Term Memory (LSTM) algorithms. Finally, attention is also paid to events that are not the result of social behaviours, but of natural or human-caused disasters [3].

Table 1 provides an overview of existing state-of-the-art approaches in special events detection and a comparison to the proposed approach, considering several criteria. Most techniques cover one or two specific types of events, e.g., social events [25] or emergency events [3], while our approach covers a broader variety of detected events, ranging from social and national events to emergency situations and network outages. Furthermore, very few approaches of the state-of-the-art tackle the classification of special events in addition to detecting them. Our algorithm relies on per-app profiles and spatiotemporal group analysis for that purpose. Finally, our algorithm has a polynomial space and time complexity, which is less than other approaches using Deep Learning (DL) techniques.

2.2. Per-app mobile traffic analysis

Hereafter, we review the literature on mobile traffic analysis covering per-mobile app analysis.

Up to our knowledge, in the literature, the detection of anomalies such as special events is not tackled at the app level yet. We believe that analysing per-app usages can give valuable details about the nature of events, and thus can help finely characterise them. In the literature, attention is paid to the app usage for other purposes than special events detection [16, 26, 35, 36]. In [26], authors provide an analysis of spatiotemporal heterogeneity in nationwide app usage - they notice a large bias between apps (even within the same category) that makes the time series clustering inconclusive, and some heterogeneity even when looking to activity peaks of individual apps. Authors in [36] investigate the similarities and differences

across different apps; as a result, they identify several well-differentiated clusters for each category of apps. In [35], the authors design a system able to identify key patterns of cellular tower traffic by clustering custom pattern identifiers in traffic into five categories: resident, transport, office, entertainment, and comprehensive, area. They study time and frequency-domain representations for traffic modelling by analysing interrelationships between traffic patterns and using a frequency-domain Fourier transform-based analysis. [16] provides a complete comparative evaluation of the techniques for signature classification, including Weekday-Weekend, typical week, median week, etc. Results unveil the diversity of baseline communication activities across countries but also evidence the existence of a number of mobile traffic signatures that are common to all studied areas and specific to particular land uses.

2.3. Group anomaly detection

While anomaly detection typically regards data point anomalies, group anomaly detection seeks to detect anomalous collections of points. Traditionally, Seeded Region Growing [2] has been used in image processing to form regions into which the image is segmented, by grouping seeds (i.e., either individual pixels or regions). The Mixture of GMM uses topic modelling for group anomaly detection. Adaptive topics are useful in recognising point-level anomalies, but cannot be used to detect anomalous behaviour at the group level. More recently, [34] studies the group anomaly detection problem by discovering anomalous aggregated behaviours of groups of points. They propose the Flexible Genre Model, which is able to characterise groups' behaviours at multiple levels, contrary to traditional topic models. This detailed characterisation enables the detection of various types of group anomalies. [11] performs group anomaly detection with an emphasis on irregular group distributions. The authors formulate two deep generative models for group anomaly detection.

Other approaches specifically focus on spatiotemporal outlier detection. In [19], the authors review outlier detection for spatiotemporal data, among other things. Considering the temporal and spatial neighbourhood for detecting outliers, they define spatiotemporal outliers as spatiotemporal objects whose behavioural attributes are significantly different from those of other objects in their spatial and temporal neighbourhoods. They propose a typical spatiotemporal-outlier detection pipeline. Many approaches leverage clustering to compute spatial outliers [7, 12]. Others use distance-based outlier detection and Voronoi diagrams to establish spatial clusters [1].

2.4. Our contribution

With respect to the above described related work, our solution does also cover the detection of so-called group anomalies, besides possible infrastructure-related anomalies. In addition, contrary to most similar approaches, our solution finely characterises the identified events through per-app profiles, for which we are not aware of specific literature at the state of the art, in particular looking for a typology of impacted apps mapped to a given category of events. Relatively to group anomaly detection, we use a two-step algorithm to detect abnormal spatiotemporal events, i.e., groups of anomalies spatially and temporally close. We first group them spatially by applying a recursive algorithm by region growing as the first approaches of the state of the art did. To do so, we prior neighbours as adjacent Voronoi cells, then identify abnormal cells as those with many anomalies. We then join spatial groups temporally with a custom algorithm. Consequently, our algorithm has a polynomial time complexity, which is less than other approaches from the state-of-the-art, especially compared to deep learning-based approaches such as LSTM networks [32].

3. Measurements and dataset

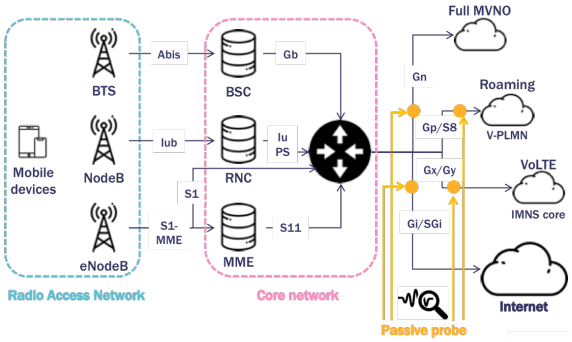


Figure 1: Simplified 2G/3G/4G network with passive probes.

Our dataset was collected at the mobile core for user sessions by Orange, a major European mobile operator, in the frame of the CANCAN project [9]. This data is collected at the Gateway GPRS Support Node, which allows knowing the app used during an Internet connection (e.g., mail, video streaming, VoIP, gaming) and the consumed data amount. Fig. 1 shows all the capture points of the probe; in our collection, we monitor Orange France traffic only, i.e., seen at Gi/SGi and Gx/Gy interfaces. Raw data is organised in TCP sessions with timestamps and cell locations of the start and end of the session. We aggregate the TCP sessions per 30-minute timeslots, for each Voronoi cell (composed of co-located base stations), each app, and each feature. Our final dataset is thus composed of a set of per-app per-feature per-cell time series.

The dataset describes the mobile traffic generated by the Orange subscriber base in Ile-de-France, the region centred on the capital Paris. After filtering underused stations, i.e., active during less than 10% of all timeslots during the whole period, the region that we analyse covers 2142 base stations sites, each including usually at least three base stations. It covers almost 3 months, from Mar. 16, 2019, to June 6, 2019. The time frame allows capturing the vast majority of mobile traffic dynamics, which

are known to occur over weekly timescales, while avoiding that the dataset size becomes unmanageable.

Ethics. Note that our study does not breach user privacy or raise ethical or legal issues. On the one hand, we never processed individual or personal data, as raw data is aggregated at the antenna level, which ensures that application demands are merged over several hundreds of subscribers. On the other hand, all data used in the analysis classify as secondary use data; i.e., the data were not collected by the mobile network operator specifically for our study, but prior to it under the control of Orange Data Privacy Officer and in compliance with applicable regulations.

4. *ASTECH* Methodology

In this section, we introduce our methodology to detect group anomalies from mobile app usage data, which we name *ASTECH* (Anomaly SpatioTEmporal Convex Hull) detection.

4.1. Algorithmic approach

For the sake of presentation, let us illustrate our methodology with the reference map of the space covered in our tests, the Ile-de-France area (Paris suburbs); this region contains various land uses including residential areas, employment areas, tourist areas, and recreation areas. We compute first the Voronoi tessellation from the list of the antennas' coordinates. Fig. 2 depicts the Voronoi tessellation of the studied area. The colour of each cell of the diagram varies with the number of packets during the 3-month period that we study, from yellow for the smallest number of packets to purple for the largest one.

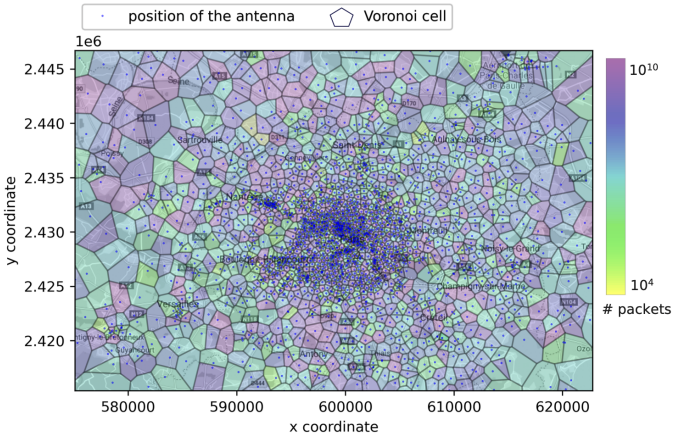


Figure 2: Voronoi tessellation of the Ile-de-France area centred on Paris. The colour gradient represents the number of packets accounted for in the cell over the 3-month period.

Different variables and dimensions characterise spatiotemporal mobile app usage data: besides space and time interval (structured in Voronoi cells and 30' slots in our setting), based on which one can aggregate both temporally and spatially, apps can be grouped together, and that as a function of multiple features. In order to scale with these dimensions while taking into consideration useful variables for anomaly detection, we propose the following algorithmic 3-stage approach:

- **Step 1**: Time series decomposition, extraction of residual component, and anomaly detection;
- **Step 2.1**: Aggregation of multi-source anomalies into snapshots;
- **Step 2.2**: Detection of group anomalies via spatiotemporal grouping of abnormal snapshots;
- **Step 3**: Classification of group anomalies into several categories through feature-based clustering.

Fig. 3 depicts the group anomaly detection methodology steps as well as its four components. We then describe each step in detail in the following two sections. We describe hereafter and in Table 2 the notations that we use in the paper.

Table 2: Notations

Variable	Description
Φ	Set of n distinct base stations
$V(\Phi)$	Voronoi diagram of Φ
F	Set of features
A	Set of apps
T	Set of timeslots
$y^f_{c,a}$	Time series for feature f, Voronoi cell c and app a
$\mathcal{R}^f_{c,a}$	Residual component of time series $y^f_{c,a}$
$\mathcal{Y}$	Set of original time series
$\mathcal{R}$	Set of residual components from original time series
$\mathcal{L}$	Set of local anomalies
$\sigma_{c,t,*}$	Snapshot of cell c at t and traffic variation $*$ in $\{+,-\}$
Γ	Set of group anomalies

4.2. Notations

Let Φ be a set of n distinct base stations (BSs) on the Euclidean place $\mathbb{R}^2$. At a given BS site, there are in practice several co-located BSs that cover different frequencies and especially 2G, 3G, and 4G: we assimilate such a group of co-located BSs to one single BS site in this work. Let $V(\Phi)$ denote the Voronoi diagram from Φ.

Let T be the set of timeslots[2], and A be the set of apps that can be associated with each session by the provider. We compute features on a per-app ($a \in A$) and per cell ($c \in V(\Phi)$) basis; the features f in F we use are: the number of users; the uploaded traffic volume, measured as the total number of uploaded packets; the downloaded volume, measured as the total number of downloaded packets.

In our tests, thousands of mobile apps are used; to limit the number, we selected the top-40 apps totalling 80% of the traffic volume. Traffic volumes generated by mobile

[2]In our tests, we use 30-minute timeslots in the period from Mar. 16 to June 6, 2019., where $n = 4320$.

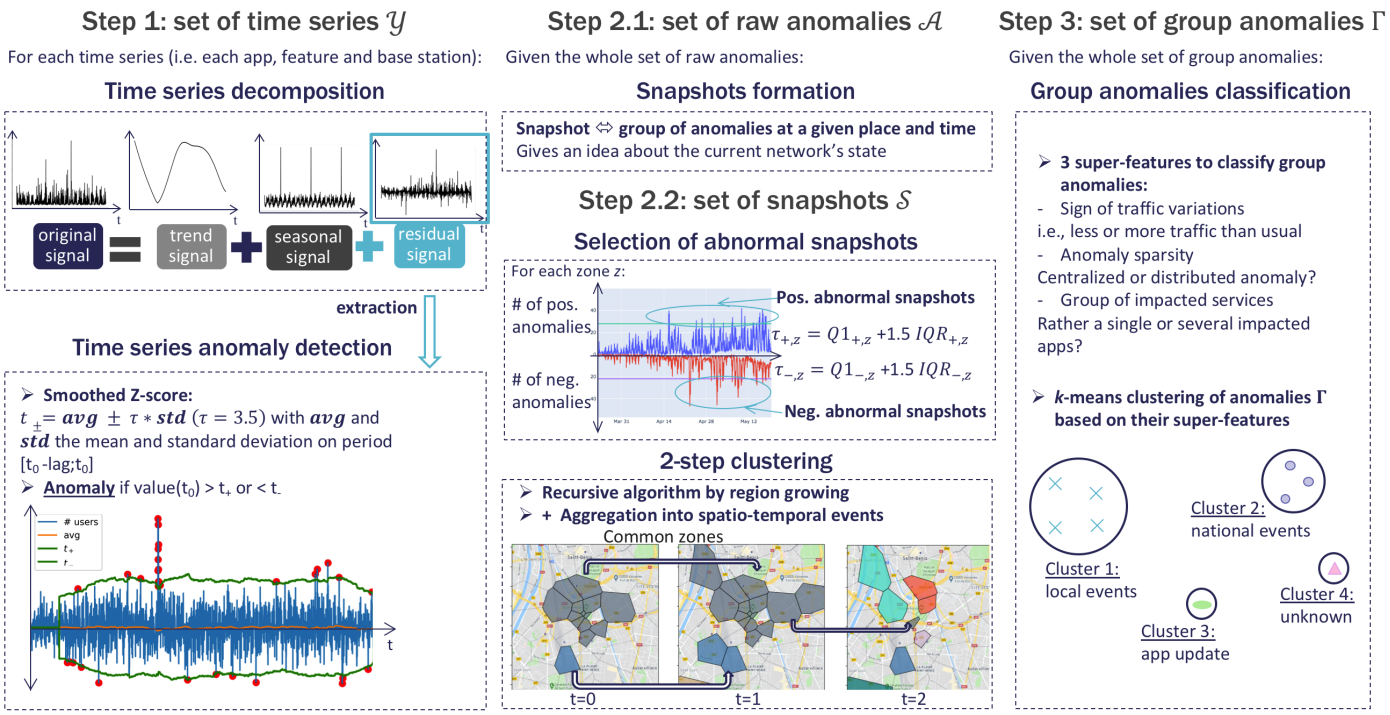


Figure 3: *ASTECH* processing steps. **Step 1**: collection of time series, one for each unique app, feature, and Voronoi cell; extraction of the residual component over which a change point detection (z-score) algorithm is applied to collect sudden time series changes leading to a set of anomalies. **Step 2.1**: anomalies are aggregated into snapshots, then **Step 2.2** only the most abnormal ones are aggregated into spatial groups, which are in turn grouped into spatiotemporal groups that we call *group anomalies*. Finally, **Step 3** provides the categorisation of such group anomalies through three super-features, by using the k-means clustering algorithm.

apps follow a power-law [31], hence only a very limited set of services yield considerable demands that are worth investigating. The set of 40 mobile apps that we get encompasses video and audio streaming (e.g., YouTube and Spotify), social media (e.g., Facebook), messaging (e.g., WhatsApp), stores (e.g., Google Play), navigation (e.g., Google Maps and Waze), as well as traffic generated by generic digital activities (e.g., web browsing, newspapers, and email).

5. Time series anomaly detection

We further develop our methodology to detect raw anomalies from feature-based time series. We first decompose time series into several components (*Sect. 5.1*). We then process the residual component derived from the decomposition to detect raw single-ton anomalies (*Sect. 5.2*).

5.1. Time series decomposition

For each app $a \in A$ and each Voronoi cell $c \in V(\Phi)$, we denote the time series of feature $f \in F$ as: $y^f_{c,a} = \{y^f_{c,a,0}, y^f_{c,a,1}, ..., y^f_{c,a,n}\}$. Let $\mathcal{Y}$ be the set of time series y, defined for each feature f, Voronoi cell c and app a.

$$\mathcal{Y} = \{y^f_{c,a}\}, \forall f \in F, \forall c \in V(\Phi), \forall a \in A \quad (1)$$

The decomposition of time series [17] is the process of deconstructing a time series into several components, each representing one among many possible underlying pattern categories. A trend component and a cycle component are usually combined into a single trend-cycle component (often called the trend). We adopt a conventional set of three components, as follows:

- $\mathcal{T}^f_{c,a}(t)$, the trend-cycle component at time t, which reflects the long-term progression of the series (the secular variation), while the cyclical component reflects repeated but non-periodic fluctuations.
- $\mathcal{S}^f_{c,a}(t)$, the seasonal component at t, reflecting seasonality (seasonal variation). A seasonal pattern exists when a time series is influenced by seasonal factors. Seasonality occurs over a fixed period (e.g., the day of the week, the hour of the day).
- $\mathcal{R}^f_{c,a}(t)$, the residual component at t, which describes random, irregular influences. It represents the residual or remainder of the time series after the other components are removed.

Two types of decomposition are commonly used. The additive decomposition is $y^f_{c,a}(t) = \mathcal{T}^f_{c,a}(t) + \mathcal{S}^f_{c,a}(t) + \mathcal{R}^f_{c,a}(t)$ while the multiplicative decomposition is written as $y^f_{c,a}(t) = \mathcal{T}^f_{c,a}(t) \times \mathcal{S}^f_{c,a}(t) \times \mathcal{R}^f_{c,a}(t)$.

The additive decomposition is appropriate if the magnitude of the seasonal fluctuations does not vary with the level of the time series. When the variation in the seasonal

pattern appears to be proportional to the level of the time series, then a multiplicative decomposition is more convenient [24]. In our case, we use the additive decomposition because we do not experience a strong trend-cycle component that would significantly amplify the whole signal.

Let us review the main techniques for time series decomposition, to then justify the one we use.

The classical time series decomposition method originated in the 1930s, and widely used then, is the *Moving Average (MA)* one [17]. It is used as the basis of many time series decomposition methods. This technique does not fit well data containing outliers. The trend component is computed as the moving average over the time series, then in the case of an outlier, the trend gets very high during a whole sliding window after the beginning of the outlier. This may perpetrate false positives, i.e., a very high amount of traffic, then an artificial drop in traffic (see more details in subsection 7.1). Other time series decomposition techniques are those using month-granularity metrics such as X11 [24] and SEATS [20].

Finally, another time series decomposition technique is the *Seasonal and Trend decomposition using LOESS (STL)* [14]. LOESS stands for LOcally Estimated Scatterplot Smoothing. This method estimates nonlinear relationships by combining multiple regression models based on k-nearest-neighbour models. It fits simple models, such as linear least squares regression, to localised subsets of the data that confer the flexibility of nonlinear regression. Taking into account the locality thus describes the deterministic part of the variation in the data, point by point. It presents several advantages over the MA, X11, and SEATS decomposition methods:

(i) Unlike X11 and SEATS, STL handles any type of seasonality, not only monthly and quarterly data.

(ii) The change rate for the seasonal component as well as the smoothness of the trend-cycle can be chosen by the user.

(iii) Contrary to other methods, STL can be robust to outliers, so that unusual observations do not affect the estimates of the trend-cycle and seasonal components. This alternative STL version uses LOWESS [15] (Locally Weighted Scatterplot Smoothing), which re-weights data when estimating the LOESS using a data-dependent function. Using the robust estimation allows the model to tolerate larger anomalies in the original signal.

We apply the robust *STL* decomposition to the $y^f_{c,a}$ time series with the periodicity set to $7 \times 48 = 336$ (for 7 days multiplied by 48 30' timeslots in a day), to take into account the hourly and daily seasonality occurring during a week. We then retain the residual component $\mathcal{R}^f_{c,a}$ in order not to be influenced by seasonal and trend variations. The set of residual components computed from the set $\mathcal{Y}$ of time series is written $\mathcal{R}$. Using the residual component rather than the original one can avoid seasonal anomalies, e.g., during rush hours, and also accentuate anomalies when framed in their context. We develop such benefits in Sect. 7.1.

5.2. Detection of raw anomalies

We detect the activity peaks and drops in the $\mathcal{R}^f_{c,a}$ time series using the *z-score* algorithm [22][3]. It compares the original signal versus its z-score, and tags elements whose absolute values are greater than the threshold as anomalies. The algorithm exploits the principle of dispersion: if a new data point is a given x number of standard deviations away from some moving mean, it is marked as an anomaly. It takes three parameters as inputs: the **threshold** τ, i.e., the z-score at which the algorithm produces an anomaly; the **lag** l, i.e., the number of past samples in a one-week sliding window. τ is set to 3.5 as recommended in [22]. As we use time series with a 30' granularity in the weekly seasonality, the lag is equal to $48 \times 7 = 336$.

Let $Z(t)$ be the z-score at time t computed as:

$$Z(t) = (|\mathcal{R}^f_{c,a}(t)| - \overline{\mathcal{R}^f_{c,a}})/\nu, \tag{2}$$

with $\overline{\mathcal{R}^f_{c,a}}$ and ν respectively the mean and standard deviation computed over the list of $\mathcal{R}^f_{c,a}$ values from $t-1-l$ to $t-1$. If $|Z(t)|$ is strictly greater than τ, an anomaly can be denoted by the 5-tuple $(t, c, f, a, Z(t))$ as composition of several attributes: its intensity equal to $Z(t)$ that can be positive or negative (i.e., there is respectively an increase or a drop in the given app usage), the timeslot t, the Voronoi cell c, the feature f, and the app a. To solve the equation, the algorithm works on a time-sliding window basis: at each timestamp t, the mean and standard deviation over the l last values are updated. Then, an anomaly is output is the absolute value of the z-score exceeds τ.

We refer to **positive** anomalies as anomalies with a positive intensity, e.g., $Z(t) > \tau$ and to **negative** anomalies as anomalies with a negative intensity, e.g., $Z(t) < -\tau$.

Note that we handle time series only above a given number of non-null samplings. For example, some cells (like those covering the stadium stands) are active only when an event occurs. If there are too few values in the time series, the learning period is not relevant and this may produce false positives. We then apply the z-score only when the learning period contains at least 30 values (out of the 336 timeslots during one week), corresponding to 15 hours coverage on the week-wide window. We chose 30 as the least value because, the sides cases with 30 quasi-consecutive non-null samplings (15 hours) can cover one full day, or two long evenings, which we esteem sufficient enough for detecting only major anomalies lasting hours. By applying this filter, we go from 848,688 to 92,940 anomalies, eliminating those that were identified as anomalies because of too few values in the time series.

The set of raw anomalies $\mathcal{L}$ found by applying the z-score on the set of residual time series $\mathcal{R}$ is thus:

$$\begin{aligned} \mathcal{L} = \{(t, c, f, a, Z(t)) \; \forall t \in T \text{ if } |Z(t)| > \tau\}, \\ \forall \, \mathcal{R}^f_{c,a}(t) \in \mathcal{R}, c \in V(\Phi), f \in F, a \in A \end{aligned} \tag{3}$$

[3] Implementation available at https://gist.github.com/ximeg/.

We refer to the elements of $\mathcal{L}$ as raw anomalies because related to a single app and not yet grouped, which is covered by the next steps.

6. Group anomalies

In this section, we present our methodology to detect group anomalies, meant as groups of raw anomalies that are spatiotemporally adjacent, i.e., they form what we call a 'convex hull'of anomalies adjacent in time and space. First, we pass through a grouping into 'snapshots', then grouped in turn spatially and temporally to form group anomalies.

6.1. Identification of abnormal snapshots

A snapshot is defined as a group of anomalies pertaining to the same timeslot t and Voronoi cell c. We distinguish between snapshots of positive and negative anomalies, i.e., the anomalies whose z-score is respectively positive and negative. A snapshot $\sigma_{c,t,*}$ is thus defined by four elements:

- the Voronoi cell c;
- the timeslot t (30-minute in our tests);
- the operator $*$ in $\{+,-\}$, respectively for positive and negative anomalies;
- the set of impacted apps denoted $\Sigma_{c,t,*}$.

Our objective is to identify "abnormal snapshots", we mean those with an abnormally high number of anomalies. To identify extreme values in our dataset, we use the Interquartile Range (IQR) on the snapshot cardinality. Above a threshold we set to the conventional 1.5 times the IQR, we spot snapshots with an abnormally high number of anomalies in a given cell as abnormal ones.

6.2. Detection of group anomalies

Our objective is to identify group anomalies, defined as a group of abnormal snapshots in a spatiotemporal convex hull, that is a compact spatiotemporal area of adjacent cells and timeslots: we group together abnormal snapshots that are close in terms of space and time. We propose a 2-phase grouping process: *(1)* group nearby abnormal snapshots at the same timeslot to create spatial groups, then *(2)* group spatial groups which are temporally close in order to form spatiotemporal events, the so-called **group anomalies**.

Phase 1: Creation of spatial groups. The objective is to recursively form spatial groups of nearby abnormal snapshots at timeslot t. The function GET_SPATIAL_GROUPS starts from an abnormal snapshot, then applies a **region growing** algorithm that recursively inspects all of the cells in its neighbourhood which are also abnormal, then their own abnormal neighbours, etc, until there are no more abnormal neighbouring cells to study. The function in charge of recursively finding all of the recursive neighbours of a cell is denoted GET_NEIGHBOURS. It takes into account the set of abnormal snapshots $S_{t,*}$ at t of sign $*$, the neighbours to study $\mathcal{N}$ (i.e., if they belong to an abnormal snapshot, we add them to the given group), the list *studied* of cells that have already been inspected and the dictionary $\mathcal{D}$ that contains the cells in $V(\Phi)$ as keys and their neighbours in the Voronoi diagram as values. We then repeat the operation for abnormal snapshots at t that are not yet grouped.

Phase 2: Creation of group anomalies. This phase consists of grouping spatial groups at successive timeslots into spatiotemporal group anomalies, with the function GET_GROUP_ANOMALIES. The objective is to group spatial groups at two successive timeslots that have at least one Voronoi cell in common. If a group at t does not have any cell in common with a group anomaly occurring at $t-1$, then we initialise a new group. From the set of spatial groups $\mathcal{G}$, we obtain the set of group anomalies denoted Γ. Each group anomaly $\gamma \in \Gamma$ is defined by its starting timeslot t_{start}, its end timeslot t_{end}, and the list *cells* of the sets of impacted cells, one for each intermediate timeslot.

6.3. Fine-grained characterisation of group anomalies

A precise characterisation of the detected group anomalies is needed to understand possible usages. We identified several broad categories of anomalies (including local events, national events, outages, bank holidays, app updates), however, we lack ground-truth labels. Therefore we chose to apply a clustering algorithm to the set of group anomalies, of positive and negative anomalies separately. Clustering algorithms are designed to group similar vectors into clusters and to identify isolated ones as outliers. The similarity between two vectors is commonly evaluated using a distance function like the Euclidean distance. Two vectors are defined as similar if they are close to each other, else dissimilar.

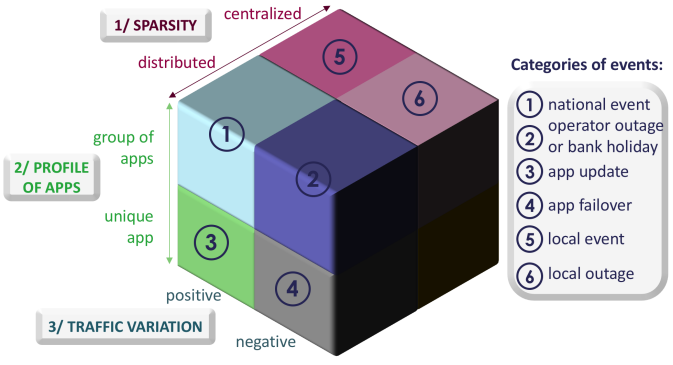
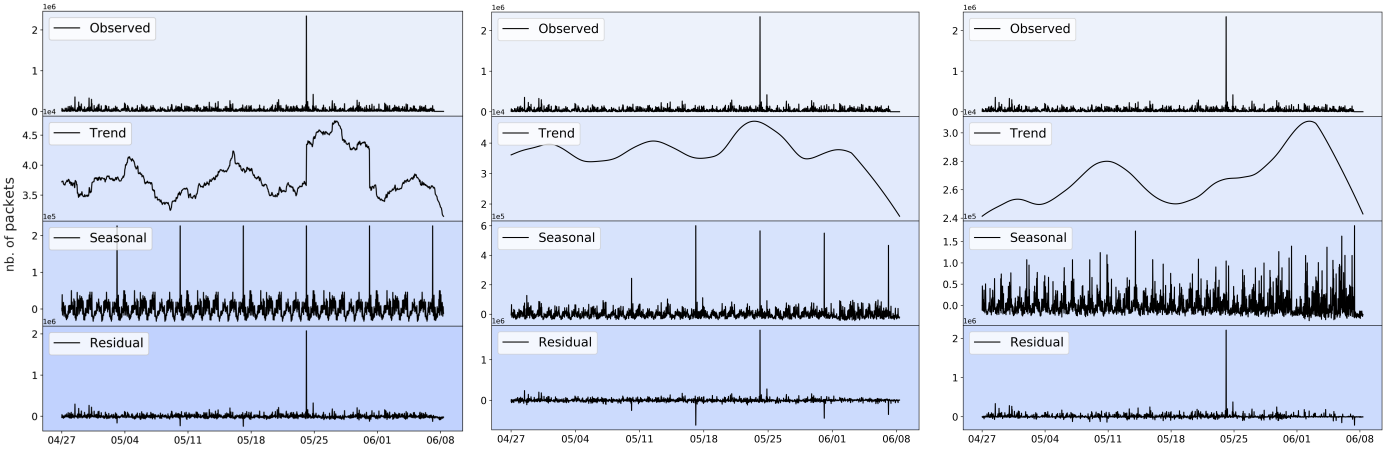


Figure 4: 3 super-features to classify group anomalies: *(1)* spatial spreading, *(2)* profile of impacted apps, *(3)* variations in mobile traffic; and 6 broad categories of events.

To cluster group anomalies, we use the k-means clustering algorithm. Unlike density-based clustering algorithms



(a) MA decomposition. (b) STL decomposition. (c) Robust STL decomposition.

Figure 5: Comparison of time series decomposition techniques: *MA*, *STL*, *robust STL* in a 6-week period.

such as DBSCAN and OPTICS which require the maximal density between data points, k-means only takes the number of clusters as a parameter, which is equal to the number of categories. Our goal is to find a match between clusters resulting from k-means and the broad categories of events that we identified.

Fig. 4 illustrates the three super-features we leveraged to classify group anomalies and the categories of events we observe. We first evaluate the group sparsity (**super-feature 1** in the figure), to determine if the group is rather localised (happening in a specific place) or distributed (happening in many far places at the same time). Then, we estimate whether the group covers rather a single app or a whole set of apps (**super-feature 2**). These two first super-features are determined by continuous variables, i.e., the group can be more or less sparse, and more or less split among a group of apps, as meant by the arrows in the figure. The last super-feature is the sign of the traffic variation (**super-feature 3**), i.e., whether it is a group of positive or negative anomalies. The list of attributes in the k-means algorithm thus includes:

- for **super-feature 1**, the weighted spatial 2D barycentre of the impacted cells for the anomaly;
- for **super-feature 2**, a one-hot encoded vector containing the 5 most recurrent apps during the given anomaly;
- we leverage on **super-feature 3** for clustering of positive and negative anomalies, separately.

Six broad categories of group anomalies emerge from the combination of super-features: local event, national event, app update, app malfunction, operator outage/bank holiday, and local outage. Note that there are eight possible combinations from the set of super-features, however we labelled only six categories as the two combinations made from "centralised / unique app / positive variation" and "localised / unique app / negative variation" were never observed and not quite relevant.

In order to get satisfying results for the clustering, we retain only major group anomalies, using ad-hoc thresholds for the anomalies' spatial spreading. Then, after the clustering, we map unclassified groups (i.e., below the threshold) to spot anomalies occurring at the same time to classify a larger range of anomalies.

7. Numerical results

In this section, we test the *ASTECH* detection methodology against data related to the Ile-de-France area for the period from Mar. 16 to June 6, 2019. We first analyse raw anomalies (related to Step 1 in Fig. 3, Sect. 5), then group anomalies (related to the other Steps, Sect. 6). Finally, we classify and characterise the group anomalies. The source code used for the detection and evaluation is available in [18].

7.1. Raw anomalies

We present results justifying Step 1 of our methodology.

7.1.1. Time series decompositions (MA, STL, robust STL)

Fig. 5 illustrates the decomposition of the time series composed of the number of WhatsApp downloaded packets over time, using three decomposition techniques: Moving Average (MA) (Fig. 5a), Seasonal-Trend Decomposition using Loess (STL) (Fig. 5b), and robust STL (Fig. 5c).

As previously discussed in Sect. 5.1, MA can be influenced by extreme values (i.e., strong anomalies) and thus may perpetrate false anomalies during a whole sliding window after a large outlier: in Fig. 5a, there is a large increase in traffic on May 23 due to a football match at *Stade de France*; then the trend component is abnormally large during one whole week after this large outlier. The seasonal

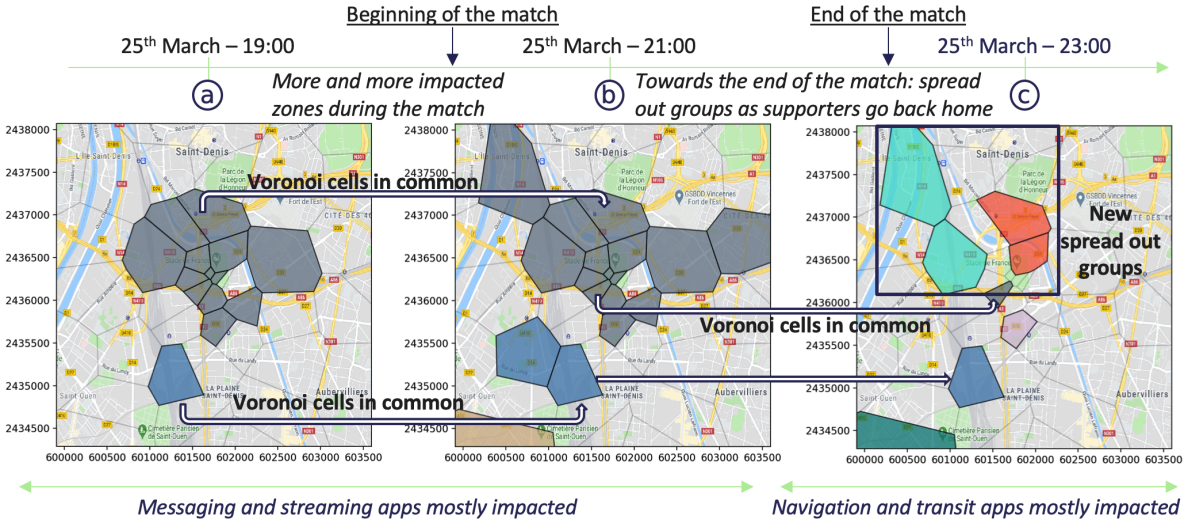


Figure 6: Timeline of the football match on March 25, 2019 in *Stade de France*. Coloured cells on the map represent abnormal snapshots at a given instant. Group anomalies are composed of spatial groups sharing at least one cell at two consecutive timeslots. Further inspection of the group anomalies gives insights about the recurring mobile applications that are impacted.

component is computed by applying a convolution filter to the data (to remove the seasonality) and by computing the average of this smoothed series for each period: in Fig. 5a, we notice that it is usually large on Saturday because of the large outlier produced by the football match on a single evening.

With STL, the seasonal component changes over time at a rate controlled by the user. Therefore, we observe in Fig. 5b that the seasonal component is impacted during a couple of Saturdays next to the Saturday of the match. The trend is also lightly impacted by the large outlier. Finally, some drops in traffic occur on Saturday evenings before and after the match in the residual series, which are false positives induced by the match.

With respect to the robust STL decomposition (using LOWESS), in Fig. 5c, the trend component is not impacted by the outlier, the seasonal component is recomputed each week and thus the residual component does not contain any false positive. Given that we observe these patterns for every time series in the dataset, as anticipated we choose the robust STL as it proved to be robust against outliers.

Component	# of pos. anomalies	# of neg. anomalies
Observed	3,256,175	0
Residual	2,293,081	799,617

Table 3: Number of positive and negative anomalies, respectively for the observed and residual components.

7.1.2. Advantages in using the residual signal

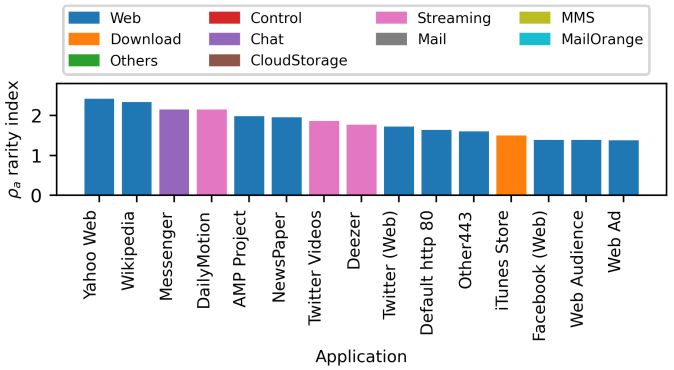
As seen in Sect. 5.1, we handle the residual component instead of the original one in our analysis. This presents several advantages. First, we avoid anomalies caused by seasonality, i.e., produced by seasonal variations like traffic peaks during rush hours and traffic drops on weekends. In addition, this enhances sudden traffic variations and makes the anomalies more visible (i.e., the z-score of the detected anomalies is even greater when using the residual component). Table 3 shows the number of positive and negative anomalies, i.e., anomalies whose z-score is respectively greater than threshold τ and lower than $-\tau$ ($\tau = 3.5$, cf. Sect. 5.2). We observe fewer positive anomalies when using the residual series rather than the observed one, as anomalies during rush hours are less visible when contextualised. Therefore, using the residual component significantly reduces the number of false positives. On the contrary, there are no negative anomalies when using the observed component, for all time series (no matter the feature, app, and Voronoi cell). This means that there is no significant drop in the original series. If we lower τ to 1.5, we notice drops at night, when there is almost no traffic anymore, but still no drops during bank holidays and massive outages. However, using the residual series enables one to emphasise these variations and thus to produce negative anomalies during outages and bank holidays. In total, we observe 799,617 negative anomalies using the residual component.

7.2. Group anomalies

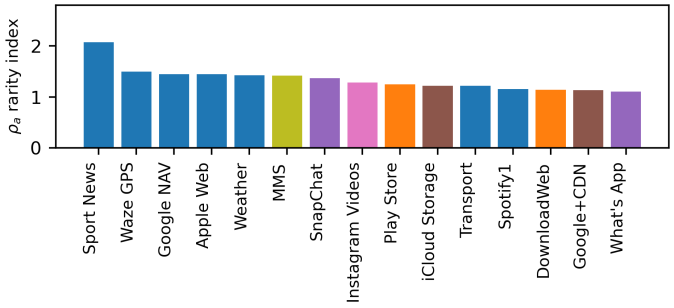
We analyse group anomalies built from raw anomalies.

7.2.1. Representation of a given event

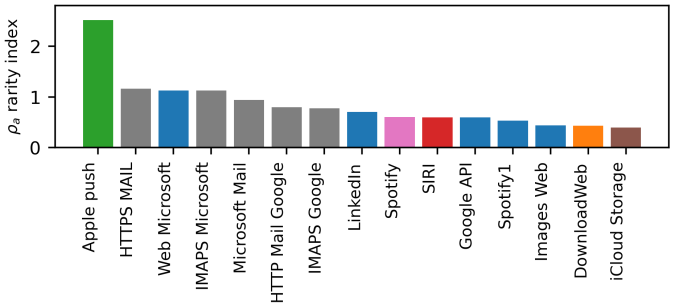
Fig. 6 represents the timeline of the football match between France and Iceland in the national stadium named *Stade de France* situated in Saint-Denis on Mar. 25, 2019, generating group anomalies. First, in **(a)**, at 19:00, before the match, we observe several impacted cells centred around the stadium. The coloured Voronoi cells represent cells whose snapshot is abnormal at that time (i.e., with an abnormally high number of anomalies). At that time



(a) For national events.



(b) For local events.



(c) For app updates.

Figure 7: ρ_a coefficient of each app for different categories e of groups of **positive** anomalies.

there are two spatial groups composed of adjacent cells of the same colour (blue and grey). The most impacted apps are messaging apps (like WhatsApp, Messenger) and streaming ones (like Instagram Videos). In **(b)**, two hours later, at 21:00, we observe spatial groups of the same blue and grey colours: they represent the evolution in time of the same two group anomalies. The number of impacted cells slightly increases as the match starts, while the nature of the impacted apps remains the same. In **(c)**, two hours later, the match is ending and supporters go back home. We then observe new group anomalies of different colours, spread out in the city and not centred around the stadium anymore. This time, the most impacted apps are navigation apps (like Uber, Google Maps).

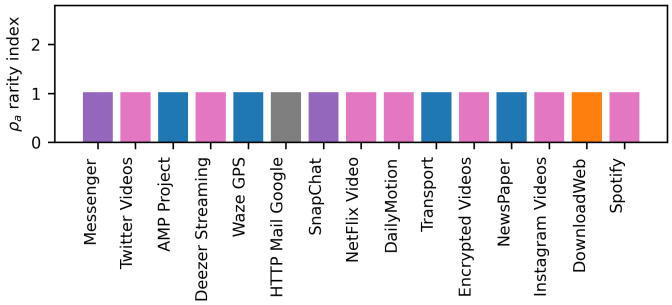
7.2.2. Typology characterisation of abnormal apps

Typology characterisation (global patterns). We now investigate whether there is a typology of the impacted apps specific to each category of events (e.g., local event, national event). The support of app a for category e is computed as the ratio of instances from e that contain at least one anomaly produced by app a. However, there may be a correlation between the app's popularity, i.e., the number of packets for the given app, and the number of anomalies produced by this app. Therefore the support of a given app may be high because the app is highly used in general (thus often produces anomalies), and not because it is highly impacted during this category of group. Therefore, instead of computing the support of app a, we define the rarity coefficient ρ_a for app a and category e of events as:

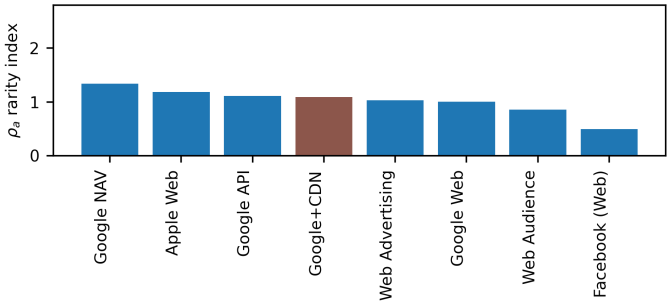
$$\rho_a = \frac{\Pr(a \in \mathcal{L} \mid e)}{\Pr(a \in \mathcal{L})} = \frac{\#\text{ events } e \text{ containing } a \;/\; \#\text{ events } e}{\#\text{ events containing } a \;/\; \#\text{ events}}. \quad (4)$$

Then, a rarity index ρ_a smaller than 1 for event e means that app a appears less frequently during event e than usual, while a rarity index greater than 1 means that a appears more frequently during e than usual. Ultimately, a rarity index ρ_a equal to 10 for event e means that app a appears 10 times more frequently during e than usual.

Fig. 7 shows the value of the ρ_a coefficient for each app a within each category of groups of **positive** anomalies. The colour of each bar represents the category of app, as defined by Orange. During *national events* (Fig. 7a), a whole subset of apps is impacted, including web apps (Wikipedia, AMP, Yahoo and NewsPaper), and some streaming and chat apps. During *local events* (Fig. 7b), the subset of most impacted apps is mostly composed of web Apps (SportNews, Waze, GoogleNAV, Weather) and MMS and

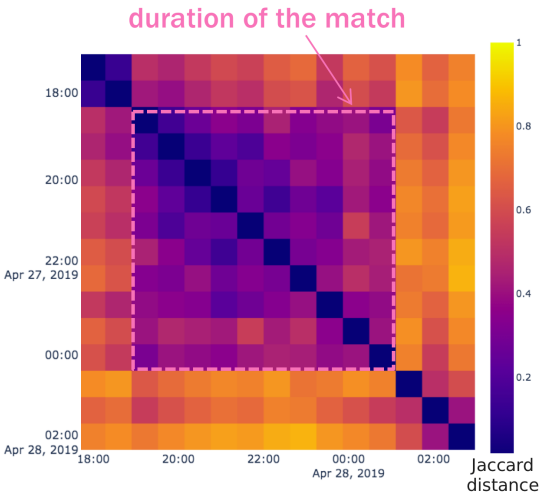


(a) For bank holidays.

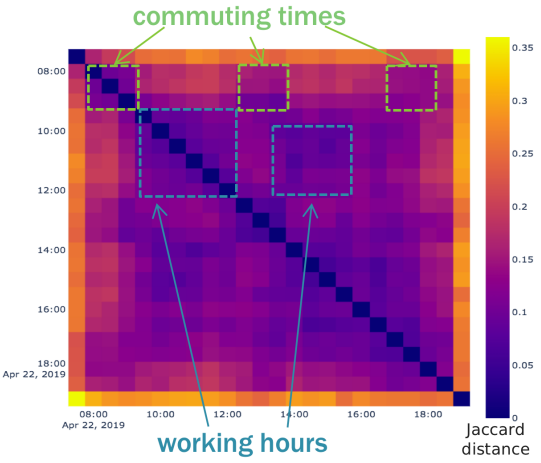


(b) For outages.

Figure 8: ρ_a coefficient of each app for different categories e of groups of **negative** anomalies.



(a) During the match on Apr. 27 in *Cornillon Stade*. The pink square highlights the heart of the match, when impacted apps were very similar.



(b) During Easter in *Saint-Denis canal*. Green squares show dark areas with similar patterns during commuting hours, while blue ones show similar patterns during working hours.

Figure 9: Heatmap of the Jaccard distances between every possible combination of two timeslots.

chat (WhatsApp and SnapChat) apps. During *app updates* (Fig. 7c), there is clearly a single impacted app which is Apple push. Therefore for groups of positive anomalies, we can induce a typology of the impacted apps for each category of groups. Local events (matches or concerts), national events (like the Notre-Dame de Paris fire), and app updates have a different signature, each one being represented by a specific set of apps.

Fig. 8 shows, for each category of groups of **negative** anomalies, the ρ_a coefficient of each app a. We observe that during bank holidays (Fig. 8a), all apps are (more or less) impacted because they are less used than usual, e.g., with a rarity index close to 1. For the two outages (Fig. 8b), there is only a small subset of impacted apps, with a rarity index close to 1. The most impacted apps are Google NAV, Apple Web, and Google API. Therefore, contrary to groups of positive anomalies, we do not notice a specific typology for groups of negative anomalies.

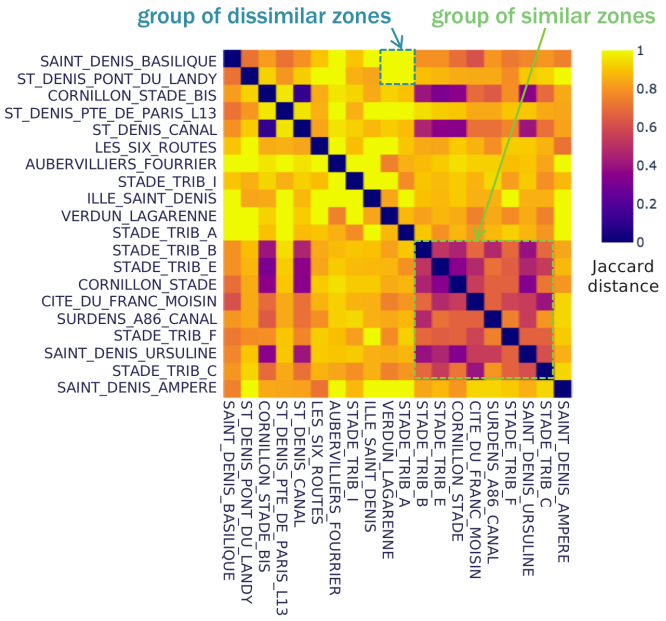


Figure 10: Heatmap of the Jaccard distances between every possible combination of two cells, on May 13, 2019 at 00:30.

Model of unpredictability (spatiotemporal dynamics of abnormal apps). We showed that there actually exist typologies of impacted apps depending on the type of events. We now study for a given event how the sets of impacted apps evolve in time and space. The Jaccard distance [23] between $\Sigma_{c,t1,*}$ and $\Sigma_{c,t2,*}$ measures the dissimilarity between the sets of impacted apps from two snapshots of a same group, at two distinct timeslots $t1$ and $t2$:

$$\text{dist}(\Sigma_{c,t1,*}, \Sigma_{c,t2,*}) = \frac{|(\Sigma_{c,t1,*} \cup \Sigma_{c,t2,*}) - (\Sigma_{c,t1,*} \cap \Sigma_{c,t2,*})|}{|\Sigma_{c,t1,*} \cup \Sigma_{c,t2,*}|} \quad (5)$$

Similarly, we can compute the Jaccard distance between the sets $\Sigma_{c1,t,*}$ and $\Sigma_{c2,t,*}$ of impacted apps from two snapshots of a same group at t, in two cells $c1$ and $c2$.

Following the evolution of such distances through time and space enables one to identify temporal and spatial patterns in the evolution of the sets of impacted apps. Fig. 9a and Fig. 9b show the **temporal** evolution of the sets of impacted apps, respectively for the match on Apr. 27 in cell *Cornillon Stade* and during Easter on Apr. 22 from 07:30 to 19:00 in cell *Saint-Denis canal*. Fig. 10 shows an example of the **spatial** evolution of the sets of impacted apps for the concert on May 13 at 00:30. The heatmaps are matrices of Jaccard distances between every possible combination of timeslots for a given cell for the **temporal** evolution, and of cells for a given timeslot for the **spatial** one. The darkest areas highlight two sets of very similar impacted apps.

We first investigate the spatiotemporal evolution of *local events* such as concerts and matches. For the **temporal** evolution (Fig. 9a), we observe that the similarity within sets of apps slightly increases, then is stable during the event (between 19:00 and 00:00), then slightly decreases at the end of the event. We find this pattern for a majority of *local events*. For the **spatial** evolution (Fig. 10), we observe that the cells located around the stadium (the ones covering the stands) are very similar to one another, while the cells farther away in the suburbs are quite dissimilar.

We then study the spatiotemporal evolution of **bank holidays**. For the **temporal** evolution (Fig. 9b), we notice an interesting pattern: first, we note some dark squares during commuting hours; the time ranges [08:00; 09:30], [12:30; 14:00], and [17:30; 19:00] exhibit very similar sets of apps. Further inspection of the recurrent impacted applications shows that it includes mostly Spotify and social networks. Then, the working hours [09:30; 12:30] and [13:00; 16:00] exhibit very similar sets of apps, with mail servers and LinkedIn mainly impacted. This heatmap shows drops in usage during Easter; meaning that these apps are less used than during normal working days. We observed this pattern for a majority of groups happening during bank holidays. For the **spatial** evolution, we do not observe any specific pattern in this case.

To sum up, we show that what our algorithm detects is exceptional behaviour from users, which we consider anomalous. We can then observe through these visualisations the extent in time and space of these types of behaviours, and for example notice localised events, their timeline, population displacement, etc.

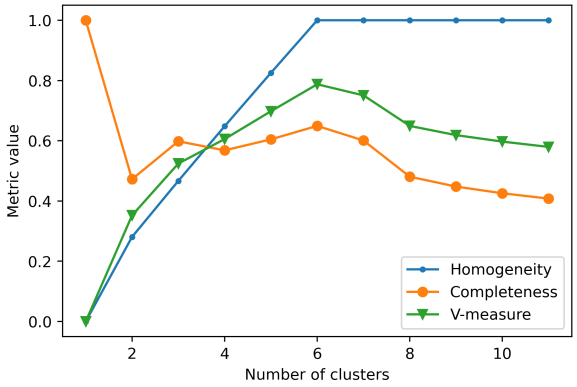


Figure 11: Homogeneity, completeness, and V-measure values depending on the number k of clusters in the k-means algorithm, for groups of positive anomalies.

7.3. Group anomalies classification

Table 4 sums up the list of prominent group anomalies that we detected in our dataset, containing mobile traffic data in Ile-de-France during the period from Mar. 16 to June 6, 2019. In total, we obtained 4,325 initial positive group anomalies and 1,063 negative ones. By selecting the top 40 % of the most spatially and/or temporally spread

Date	Event	Ground-truth label	#clus.
Mar. 24 20:00→23:00	Florence+the machine concert	local event	c1
Mar. 25 20:00→00:00	UEFA match	local event	c1
Mar. 27 20:00→23:00	Childish Gambino concert	local event	c1
Mar. 29 00:00	Brexit announcement	national event	c2
Apr. 9 16:00→17:00	Apple Push notification	app update	c3
Apr. 14 08:30→12:00	Paris Marathon	local event	c1
Apr. 15 19:00→22:30	Notre-Dame de Paris fire	national event	c4
Apr. 16 20:00→22:30	Michele Obama talk	local event	c1
Apr. 20 14:00→22:00	Hanami festival	local event	c1
Apr. 24 16:30→18:30	Apple Push notification	app update	c3
Apr. 27 19:00→23:30	French football cup	local event	c1
May 1 10:00→18:00	May Day demonstration	national event	c5
May 12 20:00→01:00	Metallica concert	local event	c1
May 24 17:00→18:00	Thérapie Taxi concert	local event	c1
May 24 17:00→18:00	Lyon bombing	national event	c5
May 26 20:00→21:00	European Parliament election	national event	c5
June 1 14:00→01:00	We Love Green festival	local event	c1

(a) Groups of **positive** anomalies.

Date	Event	Ground-truth label
Apr.15 12:00	Orange outage	outage
Apr.22 07:00→20:00	Easter	bank holiday
May 1 07:00→20:00	Labour Day	bank holiday
May 8 07:00→20:00	VE Day [a]	bank holiday
May 27 09:00	Orange outage	outage
May 30 07:00→20:00	Ascension day	bank holiday

(b) Groups of **negative** anomalies. [a]: Victory in Europe Day.

Table 4: Major group anomalies detected in Ile-de-France from Mar. 16 to June 6, 2019. Groups of positive anomalies contain events with a positive intensity, such as local events, national events, and application updates. Groups of negative anomalies contain events with a negative intensity, such as bank holidays and outages.

group anomalies, we obtain 20 prominent positive group anomalies with 15 examples shown in the table, and 64 prominent positive negative anomalies with 6 examples in the table. For each group, we indicate the date, the name, and the ground-truth label. Table 4a shows the groups of **positive anomalies**. We detected a number of local events, such as matches, concerts, talks, festivals, the Paris marathon, and demonstrations. We also identified national events like the European Parliament elections, the Notre-Dame de Paris fire, the Lyon bombing with related activity at Paris-Gare-de-Lyon in Paris, and Brexit. Finally, we detected app updates from the *Apple Push* app. Table 4b lists prominent groups of **negative anomalies**. We detected all of the bank holidays occurring during this period and in particular Easter, Labour Day, Victory in Europe Day, and Ascension Day. We also detected two outages on the Orange 4G network on Apr. 15 [33] and May 27 [30]. Note that we propose in the supplementary

materials the representation of the spatiotemporal spreading of the group anomalies we detected.

We now perform the clustering operation on the set of detected events. Given the knowledge of the ground-truth class assignments of the samples, three key related metrics reflect the quality of a clustering operation. Homogeneity is a measure of the ratio of samples of a single class pertaining to a single cluster; the fewer different classes included in one cluster, the better. Completeness is the ratio of the member of a given class that is assigned to the same cluster. V-measure is the harmonic mean of homogeneity and completeness. We choose the number k of clusters in the k-means algorithm so that the V-measure is the largest. As an example, we focus hereafter on clustering the groups of positive anomalies. Fig. 11 shows the homogeneity, completeness and V-measure values depending on k. For k set to 6, the homogeneity equals 1, the completeness 0.62, and the V-measure 0.75. Note that if we do not know the number of clusters (i.e., the ground-truth labels) in advance, we are not able to tune k. Nevertheless, the V-measure is quite high starting from 4 clusters, hence the k value is not essential to get satisfying results.

To understand the composition of the 6 clusters for groups of positive anomalies (while there are 4 ground-truth labels in our dataset), we indicate the cluster numbers assigned to the groups by k-means in the last column of Table 4a. The homogeneity equals 1, thus instances from different classes never pertain to the same cluster. However, *national events* are split into three different clusters. The first one contains the group anomaly provoked during the Brexit announcement, the second one covers the Notre-Dame de Paris fire, and the third one contains the May Day demonstration, the Lyon bombing, and the European Parliament election. These anomalies belong to three different clusters because of high variations of intensity and different typologies of anomalies impacted. Finally, the sixth cluster contains unknown anomalies that we did not identify and that do not appear in the table.

8. Discussion

In this section, we first discuss the tuning of the hyperparameters of our algorithm and their impact on the results. We then qualify the space and time complexity of *ASTECH*, considering its different steps, and discuss its scalability and applicability in real-life environments.

8.1. Tuning of the hyperparameters

Table 5 provides the list of the hyperparameters that need to be tuned, their recommended values, and a discussion on their impacts on the results. The table shows the impact of the hyperparameters at different steps of the algorithm. These settings may have an influence over the noise present in anomalies (threshold values), the sparsity of group anomalies (spatial neighbourhood degree), and the characterisation of group anomalies (k).

8.2. Complexity analysis

We qualify the space and time complexities of *ASTECH*, considering its four steps, namely time series decomposition, time series anomaly detection, aggregation into group anomalies, and group anomalies classification. The time complexity is asymptotically cubic with the number of BS, supposing the number of apps and timeslots are on the same scale as the number of BS, while the space complexity is linear with the number of detected local anomalies.

8.2.1. Time series decomposition

The STL time series decomposition is composed of three successive computations, i.e., for the seasonal, trend, and residual components.

Space complexity: the seasonal component is obtained by recombining sub-series composed of average values computed for each timestamp of the considered period, e.g., in our setting, a week. Therefore, the space complexity equals $O(\frac{|T|}{w} \times |F| \times |A| \times \phi)$, with w the number of weeks considered in T, F the set of features, A the set of apps, and ϕ the set of base stations. The trend component is computed by applying LOESS to the series without the seasonal component, thus the space complexity is $O(|T| \times |F| \times |A| \times \phi)$, with $|T|$ the set of timeslots. The residual component is computed by subtracting the seasonal and trend components from the time series, thus its space complexity is $O(|T| \times |F| \times |A| \times \phi)$. With $|F| \ll |A| \ll \phi$ and $\frac{|T|}{w} \ll 2|T|$, the asymptotic space complexity for this step is $O(|T| \times \phi)$. In our case, $|T|$ is approximately the same scale as ϕ, such that the asymptotic space complexity for this step is quadratic with the number of base stations.

Time complexity: the seasonal component has a time complexity of $O(|T| \times |A| \times \phi)$, with $|F| \ll |A|$. The time complexity for computing the trend component is $O(|T|) \times |A| \times \phi$, with $|F| \ll |A|$. The time complexity for computing the residual component is $O(3 \times |T| \times \phi)$, with $|F| \ll |A| \ll \phi$. With the values of $|T|$ and ϕ from the same scale, the asymptotic time complexity is quadratic with the number of base stations.

8.2.2. Time series anomaly detection

The z-score is then applied in a rolling window basis on the residual time series to detect outlier values. Note that steps 1 and 2 can be merged to avoid parsing twice each tuple of apps, features and base stations.

Space complexity: for each time series, one needs to temporarily store values for the mean, the standard deviation, and the z-score. Then, the list of anomalies should also be stored, thus the total space complexity is $O(3+|\mathcal{L}|)$, which can be approximated to $O(|\mathcal{L}|)$, where $\mathcal{L}$ is the set of detected local anomalies.

Time complexity: the time complexity to compute the mean, standard deviation, and z-score at each timeslot is $O(|T|)$.

Table 5: Discussion on the hyperparameters and their impact on the results.

Hyperparameter	Tuning	Impact on the results
Periodicity in STL (*Sect. 5.1*) and lag l in the z-score (*Sect. 5.2*)	One to several weeks	Higher periodicity can account for more accurate phenomena arising in cycles, e.g., seasons or school holidays.
Threshold τ on the z-score (*Sect. 5.2*)	[3, 3.5] (as in [22])	Lower threshold can account for noisier data but smaller local events detected.
Threshold on the number of anomalies in a snapshot (*Sect. 6.1*)	[1.2 * IQR, 1.5 * IQR]	Lower threshold can account for more anomalous snapshots detected.
Spatial neighbourhood degree (*Sect. 6.2*)	[1, 3]	In the current setting, only 1-neighbour spatial snapshots are aggregated. Considering higher neighbourhood degree enables to detect spatially disconnected group anomalies.
Number k of clusters in k-means (*Sect. 6.3*)	[3, 7]	Higher k can account for more specific types of group anomalies.

8.2.3. Spatiotemporal aggregation into group anomalies

Abnormal snapshots are then aggregated in space and time to form group anomalies.

Space complexity: the spatiotemporal aggregation of anomalous snapshots has a spatial complexity of $O(|\Gamma|+|\mathcal{L}|)$ to store the list of group anomalies and the list of local anomalies associated with each. The set of local anomalies being far larger than the one of group anomalies, the space complexity gets $O(|\Gamma|)$.

Time complexity: the aggregation of anomalous snapshots into group anomalies requires parsing at most twice the list of places and once the list of timeslots, i.e., the worst-case for time complexity is $O(\phi^2 \times |T|)$.

8.2.4. Group anomalies classification

Finally, group anomalies are characterised and classified using the k-means clustering algorithm.

Space complexity: the k-means clustering operation presents a space complexity of $O(k + |\Gamma|)$, e.g., to store the positions of the $|\Gamma|$ points along with their belongings to the k clusters. With $k \ll |\Gamma|$, the complexity gets $O(|\Gamma|)$.

Time complexity: the k-means clustering operation presents a time complexity of $O(t \times |\Gamma| \times d \times k)$, with t iterations, $|\Gamma|$ (d-dimensional) points, and k clusters. In the current setting, the time complexity is thus equal to $O(18 \times t \times |\Gamma|)$ accounting for 6 clusters and 3 dimensions.

8.3. Scalability and applicability in real-life environments

The asymptotic space complexity of *ASTECH* for the four steps is thus $O(\phi^2 + |\mathcal{L}|+|\Gamma|)$, while the asymptotic time complexity is $O(\phi^3)$, assuming ϕ and $|T|$ are approximately on the same scale. Our algorithm has thus polynomial time and space complexity, which is less than some approaches in the state-of-the-art, especially compared to deep learning-based approaches based on DNN (Deep Neural Network) or LSTM networks. In real-life environments, several aggregations can be considered to reduce the time execution of the proposed approach, while keeping a good detection accuracy. Both space and time complexity directly depends on $|A|$ and ϕ, which is why it is better to minimise the numbers of apps and base stations to consider. In particular, the base stations can be aggregated into zones, e.g., with clusters of Voronoi cells or grid squares, and apps into larger categories such as video and audio streaming, social media, messaging, and navigation.

9. Conclusion and perspectives

Getting a better understanding of the spatiotemporal dynamics of the group anomalies occurring in a large area can help to anticipate the load for such events. We analysed 3-month real-world mobile traffic data in which we detected transient changes in customer demand, flash crowds, and anomalies caused by unexpected crowd gathering in metropolitan areas. As a result, we detected several matches, concerts, festivals, and races, the Notre-Dame de Paris fire, elections, demonstrations, two network outages, and the four bank holidays happening during this period.

Our main contributions can be summarised as follows:

(i) We designed a methodology able to detect a number of special events in the Ile-de-France area surrounding Paris. As a result, we identified specific typologies of events, including local events, national events, app updates, outages, and bank holidays. We showed that these group anomalies can be adequately categorised through a k-means clustering operation. Insightful features include the traffic variation (positive or negative), the spatial spreading, and the impacted apps. The resulting clusters of anomalies appear to be 75% correct compared to ground-truth labels. Compared to existing works, we are able to finely characterise the type of events that may appear in a city, by also considering the set of apps that are impacted.

(ii) We showed that the apps impacted during a given event are strongly correlated to the type of event, with implications in city management, since this knowledge can be critical information on which to make decisions about events management and the prediction of services usage. Local events rather impact streaming and messaging apps, national events impact newspapers and Twitter apps, and updates impact the *Apple push* app. However, for groups of negative anomalies such as outages or bank holidays, no specific typology was identified as the anomalies consist of slight decreases in traffic compared to normal working days, thus all apps are impacted. Also, regarding unpredictability, we identified two patterns of temporal variations for the sets of impacted apps: one for local and national events where the similarity is at its peak during the

heart of the event, slightly increases, then is stable during the anomaly, and slightly decreases at the end, and the other for bank holidays composed of an alternation of commuting/break hours and working hours.

Timely detecting network anomalies allows operators to maintain the right operation of the network. Automatically detecting and characterising network anomalies allows the efficient allocation of resources. The operator can anticipate the load produced by specific types of events and provision additional resources accordingly. Also, impromptu events can be quickly detected and characterised based on their similarity to previously observed events. Areas more likely to host these events are automatically identified and tagged to be closely watched, which could allow for real-time maps exploitable for the advertisement or construction industries. Finally, knowing in advance the mobile apps impacted during given events supports efficient resource allocation.

As further work, we plan to develop an online algorithm based on our existing system detecting group anomalies in cellular traffic data. Another further work is running the analysis on a geographically larger coverage, with broader tessellation units than the base station Voronoi cell. Finally, another future work is to use noise reduction techniques in the time series in addition to time series decomposition. In [4–6], the authors introduce a technique to remove noise in the time series, by first smoothing the workload via a Savitzky–Golay filter, then adopting wavelet decomposition to decompose the smoothed outcome into multiple components.

Acknowledgements

This work was partially supported by the ANR CANCAN (ANR-18-CE25-0011) and CoCo5G (ANR-22-CE25-0001) projects. We thank Chi-Dung Phung from Cnam, Cezary Ziemlicki, and Zbigniew Smoreda from Orange Labs, for their data collection support.

References

[1] Adam, N. R., Janeja, V. P., Atluri, V., 2004. Neighborhood based detection of anomalies in high dimensional spatio-temporal sensor datasets. In: ACM SAC.

[2] Adams, R., Bischof, L., 1994. Seeded region growing. IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (6), 641–647.

[3] Bagrow, J. P., Wang, D., Barabási, A.-L., 2011. Collective response of human populations to large-scale emergencies. PLoS ONE 6 (3), 17680.

[4] Bi, J., Yuan, H., Zhang, L., Zhang, J., may 2019. Sgw-scn: An integrated machine learning approach for workload forecasting in geo-distributed cloud data centers. Inf. Sci. 481, 57–68.

[5] Bi, J., Yuan, H., Zhou, M., 2019. Temporal prediction of multi-application consolidated workloads in distributed clouds. IEEE Transactions on Automation Science and Engineering 16 (4), 1763–1773.

[6] Bi, J., Yuan, H., Zhou, M., Liu, Q., 2019. Time-dependent cloud workload forecasting via multi-task learning. IEEE Robotics and Automation Letters 4 (3), 2401–2406.

[7] Birant, D., Kut, A., 2006. Spatio-temporal outlier detection in large databases. In: ITI. IEEE.

[8] Calabrese, F., Pereira, F. C., Lorenzo, G. D., Liu, L., Ratti, C., 2010. The geography of taste: Analyzing cell-phone mobility and social events. In: Lecture Notes in Computer Science. Springer, pp. 22–37.

[9] CANCAN, 2022. Cancan project - content and context based adaptation in mobile networks.
URL `https://cancan.roc.cnam.fr/`

[10] Candia, J., González, M. C., Wang, P., Schoenharl, T., Madey, G., Barabási, A.-L., 2008. Uncovering individual and collective human dynamics from mobile phone records. Journal of Physics A: Mathematical and Theoretical 41 (22), 224015.

[11] Chalapathy, R., Toth, E., Chawla, S., 2019. Group anomaly detection using deep generative models. In: ECML PKDD. Springer, pp. 173–189.

[12] Cheng, T., Li, Z., 2006. A multiscale approach for spatio-temporal outlier detection. Transactions in GIS 10 (2), 253–263.

[13] Cici, B., Gjoka, M., Markopoulou, A., Butts, C. T., 2015. On the decomposition of cell phone activity patterns and their connection with urban ecology. In: ACM MobiHoc.

[14] Cleveland, R. B., Cleveland, W. S., McRae, J. E., Terpenning, I., 1990. STL: A seasonal-trend decomposition. Journal of Official Statistics 6 (1), 3–73.

[15] Cleveland, W. S., 1981. LOWESS: A program for smoothing scatterplots by robust locally weighted regression. The American Statistician 35 (1), 54.

[16] Furno, A., Fiore, M., Stanica, R., Ziemlicki, C., Smoreda, Z., 2017. A tale of ten cities: Characterizing signatures of mobile traffic in urban areas. IEEE TMC 16 (10), 2682–2696.

[17] Gifford, C. H. P., Macaulay, F. R., 1939. The movements of interest rates, bond yields and stock prices in the united states since 1856. The Economic Journal 49 (194), 312.

[18] Github, 2021. Source code for special events detection.
URL `https://github.com/a-blaise/special-events`

[19] Gupta, M., Gao, J., Aggarwal, C. C., Han, J., 2014. Outlier detection for temporal data: A survey. IEEE Transactions on Knowledge and Data Engineering 26 (9), 2250–2267.

[20] Gómez, V., Maravall, A., 1996. Programs tramo and seats, instructions for the user. Tech. rep., Banco de España.

[21] Hussain, B., Du, Q., Ren, P., 2018. Semi-supervised learning based big data-driven anomaly detection in mobile wireless networks. China Communications 15 (4), 41–57.

[22] Iglewicz, B., Hoaglin, D., 1993. How to detect and handle outliers. In: The ASQC Basic References in Quality Control: Statistical Techniques. Vol. 16.

[23] Jaccard, P., 1901. Étude comparative de la distribution florale dans une portion des alpes et du jura.

[24] Julius Shiskin, Allan H. Young, J. C. M., 1967. The x-11 variant of the census method ii seasonal adjustment program. Tech. rep., Bureau of the Census, U.S. Department of Commerce.

[25] Marques-Neto, H., Xavier, F., Xavier, W., Malab, C., Ziviani, A., Silveira, L., Almeida, J., 10 2018. Understanding human mobility and workload dynamics due to different large-scale events using mobile phone data. Journal of Network and Systems Management 26, 1079–1100.

[26] Marquez, C., Gramaglia, M., Fiore, M., Banchs, A., Ziemlicki, C., Smoreda, Z., 2017. Not all apps are created equal. In: ACM CoNEXT.

[27] Naboulsi, D., Fiore, M., Ribot, S., Stanica, R., 2016. Large-scale mobile traffic analysis: A survey. IEEE COMST 18 (1), 124–161.

[28] Naboulsi, D., Stanica, R., Fiore, M., 2014. Classifying call profiles in large-scale mobile traffic datasets. In: IEEE INFOCOM.

[29] Satyanarayanan, M., 2001. Pervasive computing: vision and challenges. IEEE Personal Communications 8 (4).

[30] Schmid, A., 2019. Panne Orange et SFR : des problèmes sur l'internet fixe et la 4G.
URL `https://www.phonandroid.com/panne-orange-et-sfr-des-problemes-sur-linternet-fixe-et-la-4g.html`

[31] Shafiq, M. Z., Ji, L., Liu, A. X., Wang, J., 2011. Characterizing

and modeling internet traffic dynamics of cellular devices. In: ACM SIGMETRICS.
[32] Trinh, H. D., Zeydan, E., Giupponi, L., Dini, P., 2019. Detecting mobile traffic anomalies through physical control channel fingerprinting: A deep semi-supervised approach. IEEE Access 7, 152187–152201.
[33] Turcan, M., 2019. Panne chez Orange : des problèmes de connexion Internet et 4G sur toute la France.
URL https://www.numerama.com/tech/481172-panne-chez-orange-des-problemes-de-connexion-internet-et-4g-sur-toute-la-france.html
[34] Xiong, L., Póczos, B., Schneider, J., 2011. Group anomaly detection using flexible genre models. In: NeurIPS.
[35] Xu, F., Li, Y., Wang, H., Zhang, P., Jin, D., 2017. Understanding mobile traffic patterns of large scale cellular towers in urban environment. IEEE/ACM ToN 25 (2), 1147–1161.
[36] Zhang, Y., Årvidsson, A., 2012. Understanding the characteristics of cellular data traffic. In: ACM CellNet.

Agathe Blaise received her engineering degree in computer science from ISEN, Lille, France, in 2017. She received the Ph.D. degree from LIP6, Sorbonne University, Paris, France in 2020. She is currently a research engineer at Thales Communications & Security, Gennevilliers, France. Her research interests are in the field of data analysis applied to network security, and more especially intrusion detection systems, anomaly and botnet detection.

Mathieu Bouet received the Ph.D. degree in Computer Science and the Habilitation degree from Sorbonne University (formerly UPMC – Paris VI) in 2009 and 2017, respectively. He is a Research Expert in networking and communications with Thales, France, where he currently manages research activities on network softwarisation with the Networking Laboratory, Advanced Studies Department. His research interests are mainly focused on network virtualisation and network optimisation.

Vania Conan received the Engineering and Ph.D. degrees in Computer Science from Mines ParisTech in 1990 and 1996, respectively, and the Habilitation degree from Sorbonne University, Paris in 2012. He is a Senior Research Expert in networking and communications with Thales, France. He is currently the Head of the Networking Laboratory, Advanced Studies Department in Thales.

Stefano Secci is professor of networking at Cnam (Conservatoire national des arts et metiers), Paris, France. He received the M.Sc. Degree in telecommunications engineering from Politecnico di Milano, Italy, in 2005, and a dual Ph.D. Degree in computer science and networks from Politecnico di Milano and Telecom ParisTech, France, in 2009. He was associate professor at LIP6, UPMC from 2010 to 2018. His current interests cover network automation protocols and cybersecurity. Webpage: http://cedric.cnam.fr/~seccis/.