

# EigenEvent: An Algorithm for Event Detection from Complex Data streams in Syndromic Surveillance

Hadi Fanaee-T and João Gama

Laboratory of Artificial Intelligence and Decision Support (LIAAD), University of Porto  
INESC TEC, Rua Dr. Roberto Frias, Porto, Portugal  
hadi.fanaee@fe.up.pt and jgama@fep.up.pt

## Abstract

Syndromic surveillance systems continuously monitor multiple pre-diagnostic daily streams of indicators from different regions with the aim of early detection of disease outbreaks. The main objective of these systems is to detect outbreaks hours or days before the clinical and laboratory confirmation. The type of data that is being generated via these systems is usually multivariate and seasonal with spatial and temporal dimensions. The algorithm What's Strange About Recent Events (WSARE) is the state-of-the-art method for such problems. It exhaustively searches for contrast sets in the multivariate data and signals an alarm when find statistically significant rules. This bottom-up approach presents a much lower detection delay comparing the existing top-down approaches. However, WSARE is very sensitive to the small-scale changes and subsequently comes with a relatively high rate of false alarms. We propose a new approach called EigenEvent that is neither fully top-down nor bottom-up. In this method, we instead of top-down or bottom-up search, track changes in data correlation structure via eigenspace techniques. This new methodology enables us to detect both overall changes (via eigenvalue) and dimension-level changes (via eigenvectors). Experimental results on hundred sets of benchmark data reveals that EigenEvent presents a better overall performance comparing state-of-the-art, in particular in terms of the false alarm rate.

**Keywords:** Event Detection, Complex Data Streams, Tensor Decomposition, Syndromic Surveillance

## 1 Introduction

The goal of syndromic surveillance systems is to enable earlier detection of epidemics and a more timely public health response, hours or days before clinical and laboratory confirmation comes out [15]. Two kinds of events are usually required to be detected: man-made events such as bio-terrorist activities like anthrax attacks [13] and natural events such as epidemic diseases like H1N1, avian influenza, SARS, and West Nile Virus, etc. All kinds of events regardless of their type make some changes in the environment. If we somehow manage to identify such changes in the early stages we can save many lives and prevent the potential damages. The early event detection systems are developed for such purposes. In these systems, multiple streams of pre-diagnostic health records [15, 38] such as daily counts of doctor/hospital/emergency room visits, over-the-counter medication sales, work/school absences, animal illness or deaths, internet-based health inquiries are being monitored simultaneously to trace the event footprints.

Figure 1 demonstrates an example of a complex data stream in synonymic surveillance systems. As it can be seen, this system measures 16 features aggregated daily within 8 different regions. Hence, the system generates 128 time series. Our goal is to monitor this complex system and signal an alarm when something strange occurs. One straightforward approach for monitoring such system is to monitor each individual time series and then apply an anomaly detection technique (e.g. Control chart) on each. This approach, however, imposes much higher false alarm rate. Because pre-diagnostic streams of indicators are weak and noisy signals [4] and applying detectors on each individual signal results in multiple hypothesis testing problem [48]. For instance, suppose that we reject null hypothesis when the  $p$ -value  $\prec 0.05$ , for a single hypothesis test, the probability of making a false discovery is equal to 0.05.

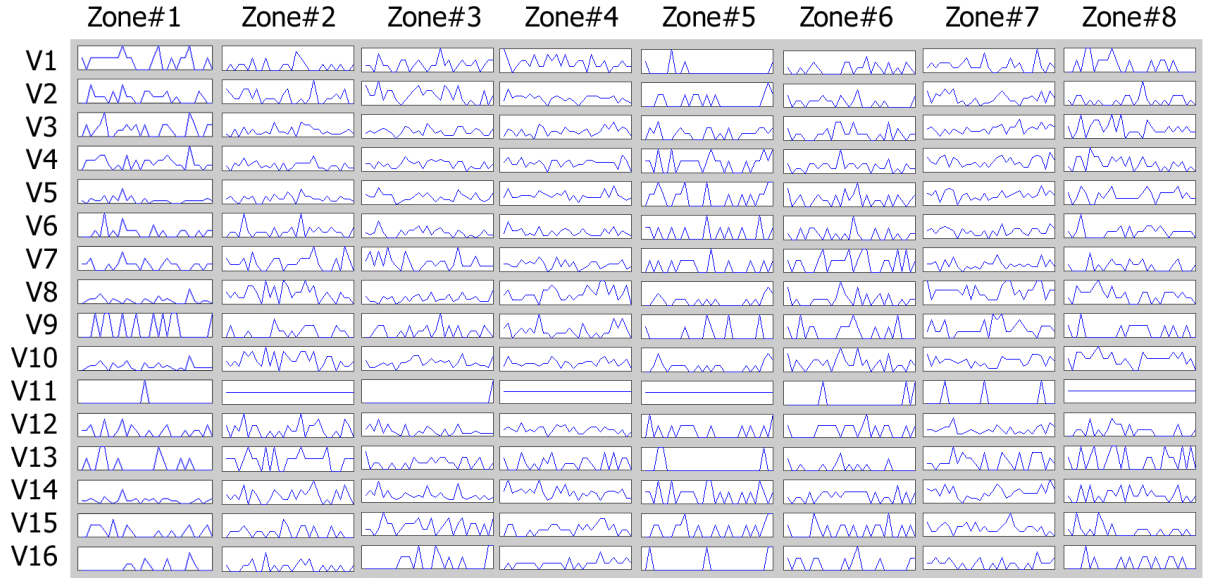


Figure 1: A sample complex system in syndromic surveillance that generates 128 time series for 16 features and 8 spatial regions.

Now assume that we do the test for each of 128 time series. Probability of false alarm could be as bad as:  $1 - (1 - 0.05)^{128} = 1.00 \gg 0.05$ .

Existing univariate methods include statistical process control based approaches [16, 46]; Time series analysis and signal processing based approaches, including singular spectrum analysis (SSA) [32], Box-Jenkins models [36]; Wavelet [50], Hidden Markov Model (HMM) [28, 35]; and regression [37]. The univariate methods since only monitor a single variable are not proper techniques for handling the complex data in syndromic surveillance. Besides, if we monitor each individual feature independently without taking into account the correlation between them, we then likely confuse the measurements error and noises with the events.

The other category of methods is multivariate methods that are able to monitor multiple streams. These methods include Hotelling T2 [49], multivariate CUSUM and EWMA [30], principal component analysis (PCA) [21], multivariate HMM [31], vector autoregression (VAR) [1, 9] and vector autoregression moving average (VARMA) [8]. There is also a sub-group of multivariate methods that operates on categorical data and looks for interesting rules via contrast set mining techniques. STUCCO [2] and Emerging Patterns [7] are instances of such techniques. Multivariate *temporal* methods, despite of their wide application in many areas, are not well-suited to syndromic surveillance and outbreak detection problems where *geographic* dimension is widely involved.

The methods that take into account geographic dimension are twofold: spatial and spatiotemporal. Spatial methods such as spatial scan statistics [22] do not capture the temporal fluctuations of the data and only operate on spatial data. Spatiotemporal methods instead take into account both spatial and temporal dimensions. Space-time scan statistics (STScan) is of this group that can operate both on univariate count data [22, 23, 24, 26] and multivariate data [25]. Univariate STScan is not adequate for syndromic surveillance for the same reason mentioned for univariate temporal methods. Multivariate STScan also has some drawbacks that make it be inappropriate for the introduced problem. On one hand they assume that the environment is static and do not consider seasonal effects and on the other hand they are developed for retrospective and offline analysis. Therefore, this group of techniques is not also suited to the problem.

There is another group of techniques such as PANDA [5] that use a causal Bayesian network to model spatiotemporal patterns of outbreaks. These methods not only explicitly compute the probability of events, but also are able to operate in real time settings through incremental updating of the Bayesian network. However, the main criticism against these techniques is that tuning the primary parameters

requires a deep prior knowledge that is not available most of the time. Therefore, these methods are considered domain specific and their application has remained limited.

Among many existing techniques and algorithms, the most suited approach to the introduced problem is WSARE [47, 48] that is able to handle multivariate data along spatial and temporal dimensions. WSARE searches for surprising rules in data streams given some baseline reference. The baseline creation strategy varies in different version of the algorithm. WSARE 2.0 uses raw historical data from selected days, WSARE 2.5 uses all historical data that match the environmental attributes and WSARE 3.0 models the baseline distribution using a Bayesian network. Opposed to PANDA where Bayesian network is created manually, WSARE 3.0 learns the Bayesian network from historical set. Therefore, is not as such domain-dependent as PANDA. WSARE has been successfully applied and merited in many real world problems such as in bioterrorism surveillance for 2002 Winter Olympics [12] and Israel influenza type B outbreak and Walerton outbreak [18]. However, the main criticism about WSARE is its high rate of false alarms [3]. WSARE opposed to other techniques, processes the data from bottom to up. Therefore, instead of overall changes in the whole data, it tracks the changes in subgroup of data. Therefore, it is sensitive to small changes and consequently presents lower detection delay, however, comes with more false alarm rate.

The methodological differences between our proposed method and WSARE are as follows. 1) WSARE is a bottom-up rule-based approach while our method is a middle approach between bottom-up and top-down that tracks both high level and dimension-based changes in the data subspace. 2) Our approach takes into account both multi-linear and multi-way correlations in data while WSARE is not able to capture such complexity; and 3) Our method is suitable only for alarming purposes and cannot explain about subgroup of the data that cause the alarm, while WSARE can be used for both purposes. 4) The statistical significance of the alarms in WSARE is computed via Monte Carlo simulation while in our approach is computed by statistical process control techniques.

In overall, the main objective in syndromic surveillance systems is to detect events in a timely manner before they turn into an epidemic. This early detection has important functions in both mortality saving and prevention of economic losses. An estimation by DARPA shows that a two-day improvement in detection time could reduce fatalities by a factor of six [33]. Another study states that improvements of even an hour in detection can reduce the economic impact of by a hundred million of dollars [44]. To reach this objective, any capable signal is required to be considered. However, this is somehow problematic, since involving more signals results in more false alarms. In the recent years the emphasis of the developed algorithms in syndromic surveillance has been focused more on the early detection and rate of false alarm is rarely taken into account. This is while the recent studies show that the false alarm rate can have an inverse effect as bad as delay in detection. A recent study concerning the warning system for tornado events [41] reveals that tornadoes occurring in the regions with a high false alarms ratio kill and injure more people. A statistically significant effect of false alarms is identified in this study: A one-standard-deviation increase in the false alarm ratio increases expected fatalities by between 12% and 29% and increases expected injuries by between 14% and 32%.

Besides, opposed to anomaly or outlier detection problems, which it is assumed that the process occurs in an isolated and static environment in syndromic surveillance systems we deal with dynamic and time-changing environment. In such environments, attributes such as day of the week, holiday, weather, etc. affects the whole or part of the system behavior. Figure 2 illustrates an individual time series corresponding to the feature  $V1$  and  $Zone1$  in Figure 1. As it can be seen, in point  $A$  due to cold weather and high rate of influenza rate, we have a higher count comparing point  $B$ . Such effects impose another kind of complexity to the event detection problem in syndromic surveillance which is required to be taken into account along with other issues.

In this paper, we propose a novel event detection methodology that considers both data complexity and time-changing environmental issues in syndromic surveillance. The concentration of this work is to reduce the false alarm rate of early event detection systems. Our contributions are as follows.

- To the best of our knowledge this is the first time that the tensor decomposition techniques [20] is applied to the syndromic surveillance problem *with space and time dimensions*.
- We use the changes in data dimensions and data correlation structure as an effective criteria for event detection.

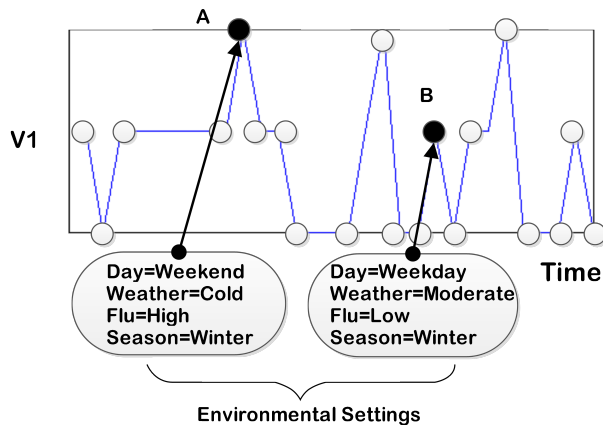


Figure 2: The environmental setting affects the data items.

- We introduce a novel and effective approach for baseline data creation that can infer baseline for unseen environmental settings.

The rest of the paper is organized as follows. In section 2 we introduce the proposed solution and our developed algorithm EigenEvent. The section 3 includes experimental evaluation, including the introduction of the data set, performance evaluation and sensitivity analysis. The last section concludes the exposition presenting the final remarks.

## 2 Proposed method

### 2.1 The idea

The fundamental idea that is used to develop the method relies on tracking changes in the subspace. This is impossible unless we could match the recent data with a *baseline reference*. However, in streaming settings, data itself is time-changing due to the effect of the dynamic environment on the data items. Therefore, using a static baseline seems to be inappropriate for dynamic environments. We propose a dynamic baseline set creation strategy which takes into account both seasonality and non-stationarity. The main novelty of our method is that we not only track changes in the feature subspace, but in the subspace of other dimensions.

Figure 3 demonstrates an illustrative example of our proposed method. Each day we receive a chunk from a complex data stream. In the data stream model this can be translated to the sliding window with fixed size of one day across the data stream. The window here is more complicated than a one-dimensional window in temporal data processing. Each window is a two-dimensional matrix of  $Space \times Features$  (top-right matrix). Each cell in the matrix corresponds to the count of a feature in specific regions. With respect to the sliding window environmental setting, we generate a dynamic baseline tensor with order of  $Space \times Features \times Time$  (top-left tensor) which is being fed from the historical data. This baseline tensor is built in each step or cycle of the algorithm run. The baseline tensor is composed of some previously arrived sliding windows that are combined in a particular order. We decompose the recent matrix and the baseline tensor to a lower-rank subspace and then match their pairwise eigenvectors and eigenvalues. We signal an alarm if we observe any unexpected difference in the match.

Figure 4 illustrates the eigenspace of both baseline and recent matrix. The solid vector in this figure corresponds to the baseline tensor. The direction of this vector corresponds to the principal eigenvector corresponding to a dimension and the length of the vector corresponds to the principal eigenvalue. When we receive a matrix we decompose it to the eigenvectors and eigenvalues and then match the obtained principal eigenvalue and principal eigenvectors to the reference vector (solid vector). We signal an alarm if the matrix eigenvector has a considerable difference in direction (eigenvector) or length (eigenvalue). For instance, dashed lines in the figure correspond to those matrices that have close eigenvector to the

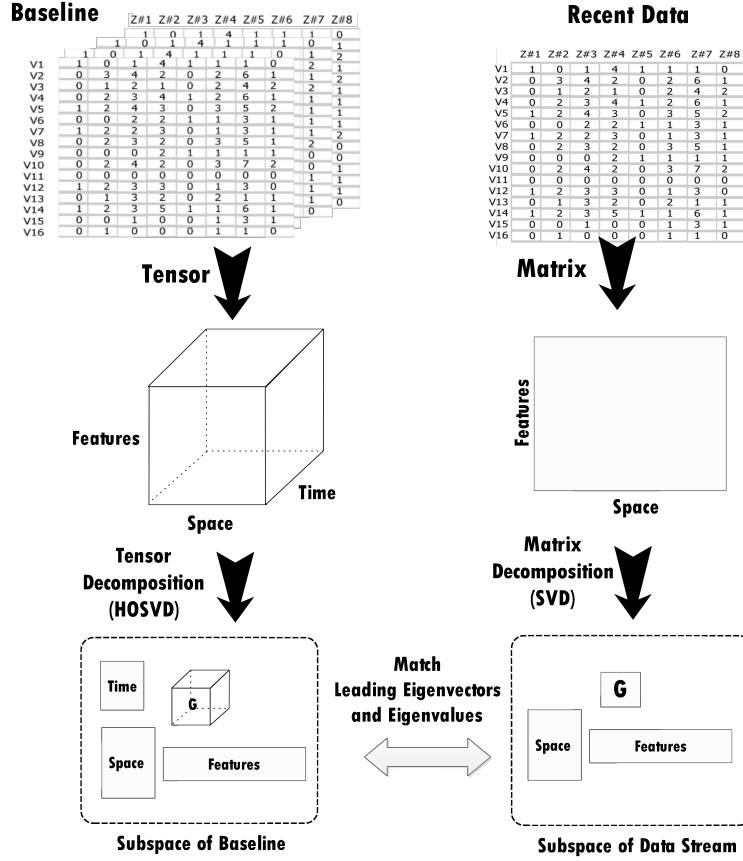


Figure 3: Snapshot of the proposed solution at a hypothetical timestamp. We detect events through tracking changes in the subspaces of baseline and recent data.

baseline eigenvector and have the close eigenvalue (vector length). Such matrices are considered normal by EigenEvent. Dash-dot lines in the figure on the contrary are related to abnormal matrices that have an unexpected eigenvector (unexpected vector direction) or unexpected eigenvalue (unexpected vector length) with respect to the baseline.

## 2.2 Proposed Algorithm: EigenEvent

In this section we describe our proposed algorithm, which is called EigenEvent. As it is presented in Algorithm 1, the inputs are as follows: sliding window  $D$  with length of one day;  $t$  which is the sequence number;  $e$  is a number corresponding to the environmental setting of the day. For instance, the environmental setting  $1214$  is related to: day=weekend(1), weather=cold(2), flu=high(1), season=winter(4). The algorithm as a result outputs a p-value indicating the statistical significance of the sliding window. A very low p-value can be interpreted as an event signal.

### 2.2.1 Data Processing and Decomposition

The first phase is to transform the sliding window to the matrix format of  $Space \times Feature$  (line 1). To assess the abnormality of sliding window we need a baseline reference to match with. Two strategies can be utilized, one is to compare the window with the previous data and another strategy is to compare the window with previous data that have the same environmental setting. We use a combined strategy that takes into account both (see section 2.2.4) and produce the dynamic baseline set according the context corresponding to the window (line 2). As a result, baseline is presented as a tensor of  $Space \times Feature \times$

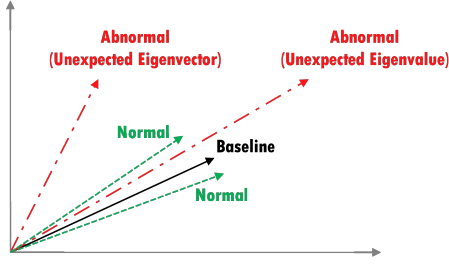


Figure 4: A simplified example showing how events can be detected by tracking changes in the eigenspace. If the distance between the sliding window eigenvector and baseline eigenvector is higher than expected, then the window is marked as abnormal. Also, if the ratio of window eigenvalue to the baseline eigenvalue is higher than expected, the window is marked as abnormal as well.

*Time.* We then apply SVD [19] on window matrix and higher order SVD (HOSVD) [6, 20] on the baseline tensor and for each dimension we take the principal eigenvector and eigenvalue.

Note that EigenEvent does not concern about the feature selection (selection of pre-diagnostic signals that are required to be monitored). Feature selection, however, may be performed via standard feature selection techniques or via domain experts or a combined technique. Nevertheless, feature selection is one the most important steps in a data mining process that is required to be taken into account. Selection of inappropriate signals may result in higher false alarm or more detection delay. The well-known over-fitting problem may happen here as well. Leinweber in an article entitle *stupid data miner tricks: overfitting the S&P 500* names some of such problems. He finds a strong correlation between butter production in Bangladesh and S&P 500 (stock market index) over a ten year period. This implies that the selection of appropriate signals still is human-dependent and cannot be fully automated.

## 2.2.2 Subspace Matching

The next phase is the matching phase. If we denote the principal eigenvalue of baseline with  $\lambda_b$ , the principal eigenvalue of window with  $\lambda_s$ , the principal eigenvector of baseline with  $X_b$  and the principal eigenvector of window with  $X_s$ , we can define the ratio of eigenvalues and Euclidean distance of eigenvectors respectively as:

$$d_{1,t} = \frac{\lambda_s}{\lambda_b} \quad (1)$$

$$\|d_{2,t}\| = (X_s, X_b). \quad (2)$$

We keep the historical distances in two vectors of  $vd_1$  and  $vd_2$  for eigenvalues and eigenvectors respectively, such that at time  $t$  we have  $vd_1 = (d_{1,1}, d_{1,2}, \dots, d_{1,t-1})$  and  $vd_2 = (d_{2,1}, d_{2,2}, \dots, d_{2,t-1})$ . Having  $d_{1,t}$ ,  $d_{2,t}$ ,  $vd_1$  and  $vd_2$  we can compute the z-scores corresponding  $d_{1,t}$  and  $d_{2,t}$  as follows.

$$z1 = \frac{d_{1,t} - \mu_{vd_1}}{\sigma_{vd_1}} \quad (3)$$

$$z2 = \frac{d_{2,t} - \mu_{vd_2}}{\sigma_{vd_2}} \quad (4)$$

Where  $\mu_{vd_1}$  and  $\mu_{vd_2}$  denotes the mean and  $\sigma_{vd_1}$  and  $\sigma_{vd_2}$  denote standard deviation of vector  $vd_1$  and  $vd_2$  respectively.

Although z-scores alone can be used along with a threshold for alarming purpose, since most related event detection algorithms in the literature outputs p-value, we may want to transform z-scores to the corresponding p-value to ease the comparison task. We can use the following equation to derive the p-value from the z-score:

$$P(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^z e^{-\frac{t^2}{2}} dt \quad (5)$$

### 2.2.3 Indicator Selection

As we already explained, HOSVD and SVD decompose the complex data into smaller subspaces (eigenspace). Tensor and matrix decomposition methods are robust against the noises. However, in the case that we have some missing values we need to use specific types of SVD [27].

We have three elements in the eigenspace that can be matched: principal eigenvector of spatial and feature dimensions and the principal eigenvalue. We may observe three kinds of changes in the match. The first kind includes an overall change in the system which is more related to the late days of outbreak period when we have both infection and outbreak. This kind of event must be reflected in a significant change in the ratio of eigenvalues ( $d_{1,t}$ ). The second kind of change occurs when an event agent (e.g. Virus) begins to spread over the geographical space. This type of event also is reflected in the changes in the spatial eigenvector pairwise distance ( $d_{2,t}$ ). The third kind is the change in the feature values. This event type can be reflected in the eigenvectors corresponding to feature dimension. However, as we show later due to the noisy properties of the feature dimension, this kind of indicator is not such helpful.

We propose a new strategy that is able to detect both overall and dimension-based changes in the system. We monitor the system using a combination of indicators, including eigenvalue and different eigenvectors and the compute the p-value corresponding to each combination for each sliding window. Then we take the minimum p-value as the algorithm output (line 9). Suppose that we have three p-values of 0.01, 0.12 and 0.43 corresponding to the pairwise match between the principal spatial eigenvector, the principal feature eigenvector and the principal eigenvalue respectively. The EigenEvent algorithm reports the minimum p-value (0.01) as the output. These above mentioned p-values indicate three facts about the system: 1) No overall change has occurred in the system, because p-value corresponding Eigenvalue is considerably high; 2) No significant change is occurring in the feature values; 3) A significant change is occurring in the spatial dimension. We may infer that data items despite of showing normal behavior in the features are showing different behavior in geographical space and hence we probably are in the outbreak phase. The minimum p-value selection strategy lets us to detect all above kinds of changes and subsequently makes the algorithm sensitive to changes in both overall system behavior and the dimension level.

### 2.2.4 Dynamic Baseline Tensor

There should be a criterion to estimate the abnormality of the recent data. As is mentioned before, two types of common criteria includes comparing with the previous data and comparing with only the previous data that match the current environment settings. Both of these criteria are vulnerable. The first criteria fails when data contains seasonal effects and second one fails when there is no enough historical data matching the recent environmental setting. To solve this problem a typical inference usually is performed, for instance, a causal Bayesian network is constructed in WSARE 3.0 [47, 48] so that when there is no enough historical data, baseline is inferred from the constructed Bayesian network. This approach, however, only make inference about the days their corresponding environmental settings cannot be found in the baseline set. In the rest of the time it compares the recent window with the previous data that match the current environment settings. This approach can be vulnerable as well, since the correlation of the current window with the recent data is ignored. We introduce another way of baseline set selection which is a combination of both ideas. We assume that the recent data is not only related to the previous data and data with the same environmental settings, but also to data with the most repeated environmental settings. In fact, our baseline tensor is a combination of previous data, data with the same environmental setting and data from most frequent environmental settings. The main advantage of this approach is that it does not fail when deal with an unseen environmental setting.

The function *BaselineTensorUpdate* in Algorithm 1 receives six inputs, including  $B$  (current baseline tensor);  $H$  (whole data, historical tensor);  $t$  (instant number);  $e$  (the recent environmental setting);  $EV$  (vector of all environmental settings seen yet); and  $C$  (recent matrix) and outputs the updated baseline tensor  $B$ . It first checks that whether the tensor  $B$  is empty. In the case that  $B$  is empty,  $C$  is added to  $B$ . Then we search in historical tensor  $H$  for data that match the recent environmental setting. Next, it rewrites the first  $k$  matrices of tensor  $B$  with the matched items.

An illustrative example of the procedure is demonstrated in Figure 5. The figure is a snapshot of the system at four hypothetical days between days 50 to 53. From the figure we also can observe four distinct environmental settings, which are shown with different colors and that their corresponding name

---

**Algorithm 1** EigenEvent

---

//D: Recent data (Right top table in figure 3)  
//C: Recent data in the format of matrix  $Space \times Features$   
//t: instant (e.g. t=3 means after 3 days of monitoring started)  
//e: Recent data Env. Setting (e.g. 1214={day=weekend, weather=cold, flu=high, season=winter})  
//EV: Environmental setting vector (e.g. [1214,1321,3214,1456])  
//H: Historical Tensor  
//vd1: vector of principal Eigenvalue distances ( $d_{1,1}, d_{1,2}, \dots, d_{1,t-1}$ )  
//vd2: vector of principal spatial Eigenvector distances ( $d_{2,1}, d_{2,2}, \dots, d_{2,t-1}$ )  
//B: Current generated Baseline tensor (Tensor  $Space \times Time \times Features$  in Figure 3)  
//P-value: Statistical Significance of the recent data (e.g. Signal an alarm when  $p - value < 0.05$ )

**Require:** D, t, e

**Ensure:** P-value

```
1: Matrix  $C \leftarrow D$ 
2: Tensor  $B \leftarrow BaselineTensorUpdate(B, H, t, e, EV, C)$ 
3: HOSVD(B):  $X_b \leftarrow principal\ spatial\ Eigenvector, \lambda_b \leftarrow principal\ Eigenvalue$ 
4: SVD(C):  $X_c \leftarrow principal\ spatial\ Eigenvector, \lambda_c \leftarrow principal\ Eigenvalue$ 
5:  $d_1 = \frac{\lambda_b}{\lambda_c}$ 
6:  $\|d_2\| = (X_c, X_b)$ .
7: p1= p-value of  $d_1$  given  $vd_1$ 
8: p2= p-value of  $d_2$  given  $vd_2$ 
9:  $P - value \leftarrow Min[p1, p2]$ 
10: if e then exists in EV
11:    $vd_1 \xleftarrow{add} d_1$ 
12:    $vd_2 \xleftarrow{add} d_2$ 
13: end if
14:  $H \xleftarrow{add} C$ 
15:  $EV \xleftarrow{add} e$ 

16: function BASELINETENSORUPDATE(B,H,t,e,EV,C)
17:   if B is empty then  $B \xleftarrow{add} C$ 
18:   else
19:     k=0
20:     for i=1 to t-1 do
21:       if EV(i) == e then
22:          $k = k + 1$ 
23:          $B(k) \leftarrow H(i)$ 
24:       end if
25:     end for
26:   end if
27:   Return B
28: end function
```

---

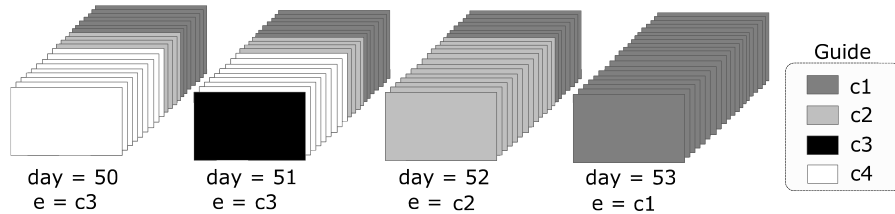


Figure 5: A Sample of dynamic baseline tensor creation process between day 50 to day 53. Each plate represents a daily matrix of  $Space \times Features$  in historical set. The dynamic baseline tensor is a combination of such matrices in a particular order.  $e$  also denotes the environmental setting of the day.



is demonstrated in the guide table. Each cube in the figure represents a baseline tensor and each plate inside the cubes is a  $Space \times Features$  matrix from the historical set. At day 50 the baseline tensor is composed of 20 matrices such that 9 of matrices are from setting **c4**, 4 matrices from setting **c2** and 7 matrices from setting **c1**. We also assume that the context **c1** is the dominant environmental setting with 20 times occurrence. The dominant context is the most frequent setting in all the history. For this reason, all baseline tensors, in Figure 5, include 20 matrices, given that the length of the baseline tensor is equal to the number of occurrences of the dominant context.

Now let’s explain how a dynamic baseline set is generated. At day 50, we receive a matrix with setting **c1**. We search in historical tensor  $H$  for a match with **c1** setting, but we do not find, so the function *BaselineTensorUpdate* returns input  $B$  unchanged. On day 51, we again receive a matrix with the setting **c1**. We again search for a match in  $H$ . This time we find one match, because one day before (day 50) the setting has been **c1**. Therefore, we rewrite the first  $k$  elements of  $B$  Tensor with  $k$  found matrices. In this case since we find only one match,  $k$  is equal to 1. At day 52 we receive a matrix corresponding with environmental setting **c2**. We search in  $H$  for a match and suppose that we find 13 matrices. Hence,  $k$  will be equal to 13, so we rewrite the first 13 elements of the baseline tensor with the matched 13 matrices. As it can be observed at day 52, setting **c2** has been the dominant setting versus **c3** and **c4** settings, however, still **c1** dominates **c2** (**c1** setting has more repeats comparing **c2**), therefore, the baseline tensor is composed of matrices with most dominant settings with preference to the recent data. Finally, on day 53, we receive a matrix with setting **c1**. We search in  $H$  for a match and we find 20 matrices ( $k = 20$ ), thus we rewrite first 20 elements of the baseline tensor with matrices corresponding **c1** settings. At this moment, the whole baseline tensor is filled with only matrices with setting **c1**. This procedure repeats and repeats. However, the size of baseline tensor always stays fixed to the repeat count of the most repeated environmental settings.

### 2.2.5 Updating Step

In this step we update the vector of distances (line 11-12). We add the distances to the vectors if their corresponding contexts has been already seen. If we have a matrix with an unseen environmental setting, we do not add the computed distance to the vector of distances. Because an inference for this setting is approximate and adding the distance obtained from this approximation is not adequate for keeping. We finally update historical tensor and vector of environmental settings.

## 3 Evaluation

### 3.1 Data set

Validation of event detection algorithms is basically a difficult task due to the type of required data [3, 39, 48]. To evaluate the algorithms, the event occurrence period is required to clearly be labeled in the data. This requires a knowledge expert to look into the data and specify the event period manually, making this task infeasible. Benchmark data sets that are already used for change detection and anomaly detection are not appropriate for our research purpose, because, on one hand, most of the time they do not have seasonality property and on the other hand do not contain multi-way property. We recently [10] proposed a semi-automatic for labeling events in unlabeled data which is based on ensemble detectors and background knowledge from web. However, this approach also needs to have access to some sort of background knowledge which is not available in this domain.

We use a benchmark data set used in [47] including 100 data sets of a simulated disease outbreak. These data sets are generated using a Bayesian network simulator namely CityBN which generates temporal fluctuations based on a variety of factors such as weather and food conditions [48]. The structure and parameter of this Bayesian network are manually adjusted. As is mentioned by the authors, this simulator produces extremely noisy data sets that are a challenge for any detection algorithm. This data set is publicly available online in [45].

Table 1 shows the characteristic of the original data sets. As it can be seen, this data is multi-way. It contains two dimensions of space and time and multiple variables. It also contains seasonal effects, because features are under influence of some environmental settings. Cardinality of each attribute is also specified in the table. As it can be seen, we have 9 distinct spatial regions and 730 temporal instants

Table 1: Characteristic of CityBN Data sets

Field	Type	Cardinality	Sample Record#1	Sample Record#2
XY	Spatial	9	SW	NE
Daynum	Temporal	730	73779	74508
Age	Feature	3	senior	child
Gender	Feature	2	female	male
Action	Feature	3	purchase	evisit
Reported symptom	Feature	4	nausea	respiratory
Drug	Feature	4	nyquil	vomit-b-gone
Flu Level in season	Environmental	4	high	decline
Day of week	Environmental	3	weekday	sat
Weather	Environmental	2	cold	hot
Season	Environmental	4	winter	sumer



Figure 6: Evaluation Strategy: Alarms in days with black color represents false alarms and in white days represents true alarms. Detection delay is also specified for each day inside the plates

(days). We also have 16 ( $3+2+3+4+4$ ) distinct time series and  $4 \times 3 \times 2 \times 4$  possible environmental settings.

### 3.2 Performance

Receiver operating characteristic (ROC) curve [14] measures the trade-off between sensitivity and specificity. ROC curve is widely used method for evaluation of anomaly detection and classification methods. However, ROC curve, even though summarizes the overall ability of the algorithm, does not evaluate the timeliness of detection which is critical in syndromic surveillance. An algorithm with the lowest false positive and the highest true positive rate that detect outbreaks with heavy delay is inappropriate for syndromic surveillance applications. In fact a system with this characteristic is more helpful for retrospective applications than the prospective applications like what is required in syndromic surveillance. One of the proper metrics for evaluation of algorithms is Activity Monitoring Operating Characteristic (AMOC) curve [11] that evaluates the trade-off between specificity (false alarms) and timeliness (detection time). AMOC curve is widely used for evaluation of methods in syndromic surveillance [4, 17, 40, 48]. Therefore, in this work we use AMOC curve for evaluation of our algorithm.

We use the same evaluation strategy as [48]. Assume that the agent release occurs at timestamp  $t$ . A true alarm corresponds to a case where the alarm is raised in a period between  $t + 1$  and  $t + 14$ . The alarms before or after this period are considered false positives. The detection delay is also defined as the temporal difference between the first alarm in the above period and the release time. In reality, the data of each day is processed tomorrow of that. Therefore, is not possible to detect event on the day of release. Thus, the optimum detection is tomorrow of the release (detection delay=1). This one day delay is also considered in CityBN simulation. Figure 6 demonstrates that how we define false alarms and detection delay. If we signal an alarm in a period of 14 days after release it is marked as true alarm and if we signal an alarm before or after this period, it is marked as a false alarm. Detection delay is also specified in the figure as numbers in the plates. If we signal an alarm tomorrow of the release, we get only one-day delay which is the optimum condition. For any alarm after this period we define detection delay equal to 14 (as [48]).

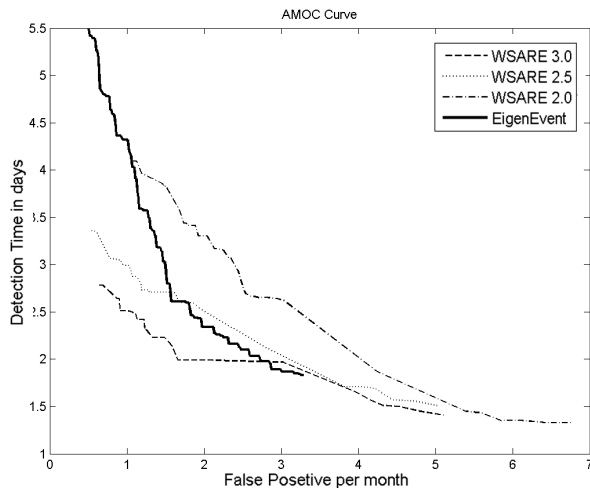


Figure 7: AMOC Curve for EigenEvent vs. WSARE

The outputs of both WSARE and EigenEvent are p-values indicating the statistical significance of recent data. Depending on the desired confidence level, we may signal an alarm. For instance, given a threshold as 0.05 we signal an alarm if the p-value corresponding to the recent data goes lower than 0.05. To assess the algorithms performances we use variable p-value threshold from 0.020 to 0.250 with the step of 0.001 (totally 231 p-values). Each data set has temporal size of 730 days. We use the first 365 days for training the primary baseline and the next 365 days for evaluation of the algorithms. Baseline set is also incrementally updated whenever a new window arrives after day 365. Note that agent release in all 100 data sets occurs in the second year and is guaranteed that the first year do not contain any release. A sliding window moves across the data from day 366 to day 730 and match each window with the baseline. If the match outputs a p-value below the threshold, then an alarm is raised. After we reached to day 730, we compute the number of false alarms and detection delays. We finally average the detection delay and false alarms for all 100 data sets and plot the AMOC Curve. In the AMOC curve, the x-axis indicates the number of false alarms per month and the y-axis measures the detection time in days. The optimal detection is one day detection delay with zero false alarm. The closer to the point (0,1) the better detection algorithm is.

The results are shown in Figure 7. Although the curve corresponding EigenEvent seems different comparing WSARE, if we rotate the AMOC curve 90 degrees anticlockwise we observe the same pattern similar to WSARE 3.0. The difference is that EigenEvent performs better in terms of false alarm rate and performs worse in terms of detection delay. The intersection between the curves makes the overall comparison difficult. For instance, in a desired false positive rate from 2.8 to 3.3, EigenEvent is the best method both in terms of false alarm rate and detection delay. Nevertheless, to specify which of the algorithms are better in overall we need to compute the area under the AMOC curve [34], average delay and average false positive rate (see Table 2). Obtained area under AMOC curve implies that EigenEvent outperforms all versions of WSARE. Its average false positive is considerably lower than all versions of WSARE. However, in terms of detection delay as was expected presents one more day delay. To have a separate look on both numbers of false alarms and detection delay, we also compute the average false alarms and detection delay for 231 p-values (from 0.020 to 0.250 with the step of 0.001). The results are presented in table 4 and 5 respectively. As it can be seen from the first table, EigenEvent in terms of false alarms, beats other methods in the majority of data sets. Regarding the detection delay even though is not the best, has detected events tomorrow of release in half of the data sets.

The main reason for the differences in the performance is related to the methodological differences between EigenEvent and WSARE. EigenEvent opposed to WSARE is not a bottom up approach and subsequently is less sensitive to the small-scale changes and subsequently, presents less false positive rate. EigenEvent due to its less sensitivity to the small-scale changes reacts slower to the events. However, EigenEvent have this ability to track changes in the dimensions, for this reason does not suffer from

Table 2: False positive rate (per month), Detection delay (in days), Area under AMOC Curve and Runtime (in seconds) Averaged for 100 data sets of CityBN

Method	False positive	Detection delay	AUAMOC
WSARE 2.0	4.052439	2.163983	12.859000
WSARE 2.5	2.739062	2.192338	9.885192
WSARE 3.0	2.877031	<b>1.929134</b>	8.648379
EigenEvent	<b>1.866439</b>	2.839827	<b>8.027842</b>

Table 3: Runtime (in seconds) Averaged for 100 data sets

Method	Runtime
WSARE 2.0	59.2
WSARE 2.5	105.3
WSARE 3.0	838.4
EigenEvent	<b>16.8</b>

the high false alarm rate problem of bottom-up approaches and heavy delay problem of the top-bottom approaches.

### 3.3 Runtime

Since in syndromic surveillance systems, data are often required to be processed in daily scale, computational efficiency receives less attention. In the unlikely case where data size becomes very huge and processing of data requires run-time of more than 24 hours (the process scale) then we have to come up with computational efficiency issues. Although, computational efficiency is not the claim in this research work, runtime in Table 3 indicates the superiority of EigenEvent over all versions of WSARE. EigenEvent requires only 16.8s to deliver the result. This is three times faster than WSARE 2.0, 6 times faster than WSARE 2.5 and 50 times faster than WSARE 3.0. The majority of this difference is related to two factors; WSARE exhaustively search the whole space while EigenEvent only tracks the changes in the correlation structure. The second factor is related to the method the approaches compute the p-value of alarms. WSARE exploits Monte Carlo simulations for computing the p-value while EigenEvent computes the p-value using statistical process control techniques which is lighter.

In each time step, EigenEvent requires to perform a tensor decomposition and a matrix decomposition. The offline tensor decomposition (OTA) [42] of the baseline tensor requires  $O(T \prod_{i=1}^M n_i)$  where  $T$  is the temporal size of the tensor,  $M$  is the order of the tensor which in our case is equal to 3 (three dimensions of space, features and time) and  $n_i (1 \leq i \leq M)$  is the dimensionality of the  $i$ th mode (reshaped matrix in dimension  $i$ ). The matrix decomposition of the recent data also requires  $O(N^2)$  for one-rank matrix decomposition. Therefore, in each step we require  $O(T \prod_{i=1}^M n_i) + O(N^2)$ . Some approximation techniques are developed for reducing the computation time of the first term. For instance, [42] proposed three different techniques including dynamic tensor analysis (DTA), streaming tensor analysis (STA) and window-based tensor analysis (WTA) that perform tensor decomposition more efficiently with much lower computation time. For instance, DTA requires computation time of  $2 \sum_{i=1}^M r_i n_i^2 + \sum_{i=1}^M n_i \sum_{j=1}^M n_j$  where  $r_i$  is the core size for each mode which in our case is equal to 1. Therefore, assuming the tensor with three dimensions (as our case study) we require only  $2 \sum_{i=1}^3 n_i^2 + \sum_{i=1}^3 n_i \sum_{j=1}^3 n_j$  which is a tremendous improvement over OTA. For low-order tensor (i.e.  $M \leq 5$ ) as is pointed out in [42], the diagonalization which is the main cost can be performed via faster approximation approaches. However, since computation efficiency is not the main concern in this part of our research we do not test all the available techniques.

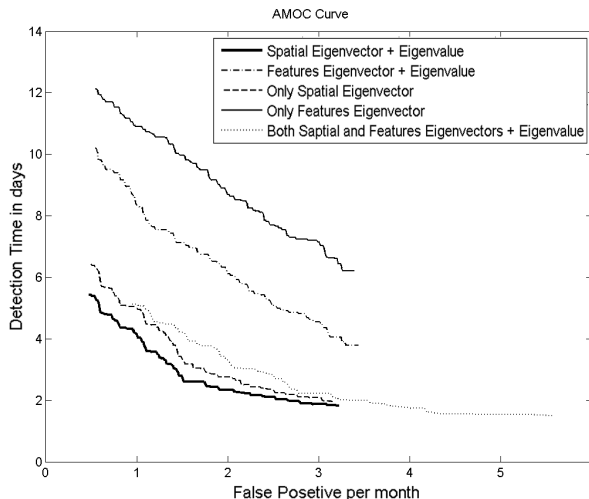


Figure 8: Effect of indicators on the performance

### 3.4 Leading Indicators

As we mentioned before, EigenEvent algorithm tracks the deviation of sliding window eigenspace from the baseline tensor eigenspace for change detection. Now the question is that what elements of the eigenspace we should take for the match. Should we opt for eigenvectors corresponding to the spatial dimension or to the feature dimension. Should eigenvalue be used along eigenvector or eigenvector alone is enough. We examine five circumstances: the first condition is the default setting in the Algorithm (The optimum selection in line 9 of Algorithm 1), and the rest are different combination of eigenvectors and eigenvalues. Figure 8 illustrates the AMOC curve for these different combinations. As it can be seen, by using only spatial eigenvector (without considering eigenvalue) we experience the same result but with more half-day average delay. In fact, involving of Eigenvalue in the change detection process provides earlier detection. We also study a condition where whole eigenspace is used. In this case we take into account both spatial and features eigenvectors along the eigenvalue. This leads to half-day delay earlier detection, but with 1.5 more false alarms. Excluding spatial eigenvector from the eigenspace matching also leads to lower performance both in terms of delay and false alarms. This result reveals that how the spatial dimension is important. In fact, temporal methods that exclude the spatial dimension lose lots of information. The reason is that feature signals are very noisy and detection of pattern of such noisy data comes with high false discovery. Instead, the spatial dimension is more stable and tracking changes in this dimension can be a better indicator for tracking particular events such as disease outbreaks (our case study), because, one of the key signatures of disease outbreak is movement in the space. This movement changes the constant patterns in the spatial dimension and subsequently this appears in the principal spatial eigenvector.

### 3.5 Baseline Selection

We compare three scenarios for baseline creation: 1) from historical set *without* respect to the environmental setting; 2) from historical set *with* respect to the environmental setting; and 3) Dynamic baseline tensor (our strategy). In the first scenario we compare the recent data with historical data without considering the environmental setting. For instance, we compare the recent data with data of last one week or the last eight weeks. In the second scenario we take reference data from the matched environmental setting of the day. For instance, if the environmental setting of recent day is 4112 we search in historical set for those  $Space \times Features$  matrices whose corresponding environmental setting is 4112. In the third scenario (our method), we create the baseline tensor from matched environmental setting, but we give more importance to the most dominant environmental setting and more recent data.

Figure 9 and 10 compares the obtained performance through these different strategies. Figure 10 illustrates the comparison of the first scenario versus the third scenario and Figure 9 compares the

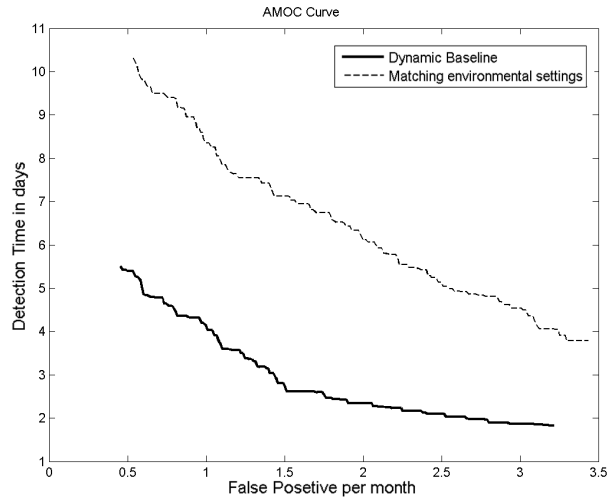


Figure 9: Dynamic baseline vs. Environmental matching baseline

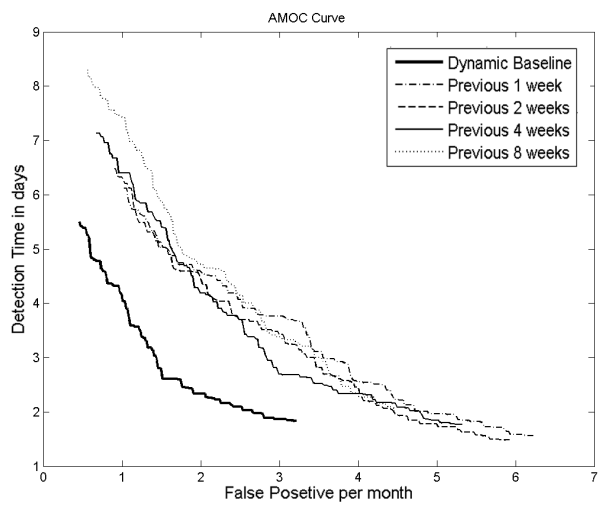


Figure 10: Dynamic baseline vs. Historical baseline

performance of the second scenario versus the third scenario. The results reveal that our dynamic tensor creation strategy outperforms the first and second scenarios. The reason of this good performance is related to this that our approach makes a better inference for unseen environmental setting. This approach is also robust to the noises and therefore provides a higher quality baseline reference.

## 4 Conclusion and future works

We propose a novel approach based on eigenspace techniques for early detection of events from complex data streams in syndromic surveillance. The purpose of this work is to reduce the false alarm rate of the state-of-the-art early detection methods. The experimental evaluation results on benchmark data sets shows that the proposed approach provides a better overall performance versus state-of-the-art. Our approach while maintains the detection delay in a reasonable level improves the false alarm rate to a considerable extent. While top-down approaches look for changes in higher level feature space and bottom-up approaches track changes in the low-level feature space, we introduce a novel methodology based on eigenspace and tensor decomposition techniques that track changes both in high level and the dimension level. The overall changes in the system appear in the eigenvalue and a change in the dimensions appears in the eigenvectors. Such dimension-based strategy is very helpful in some applications such as disease outbreak where the spatial dimension gets very important. However, using such methods makes sense when data contains further dimensions (e.g. Space and time). In other words, the competitive part of our approach is its dimension-based change tracking which is valid only for multidimensional (multiway) data.

A challenge to the future research is to utilize EigenEvent in a real-world problem and evaluate its performance in the practice. This was one of our main limitations in this research. Unfortunately, there is no public real-world data available with ground truth for syndromic surveillance research. Most of bio-surveillance programs also correspond to the governmental sections where gaining data in most of the time is impossible. Even if we access to real data, the period of outbreaks or events is not specified in that. There is a recent developed simulator [29] that simulate multivariate syndromic time series and outbreak signatures. However, since this simulator does not support the spatial dimension as the future work we are going to adapt it for this purpose and perform more experiments based on the new simulated data sets. We also intend to study the computational performance of the algorithm using incremental and streaming tensor decomposition techniques [43] which are more appropriate for large-scale data sets.

### Complements.

The MATLAB code and data sets are available online via <http://fanaee.com/research/EigenEvent>.

### Acknowledgments.

This research was supported by the Projects NORTE-07-0124-FEDER-000059 and NORTE-07-0124-FEDER-000056 which is financed by the North Portugal Regional Operational Program (ON.2 - O Novo Norte), under the National Strategic Reference Framework (NSRF), through the European Regional Development Fund (ERDF), and by national funds, through the Portuguese funding agency, Fundação para a Ciência ea Tecnologia(FCT). Authors also acknowledge the support of the European Commission through the project MAESTRA (Grant Number ICT-750 2013-612944).

## References

- [1] Barros, C. P. (2003). An intervention analysis of terrorism: The spanish eta case. *Defence and Peace Economics* 14(6), 401–412.
- [2] Bay, S. D. and M. J. Pazzani (1999). Detecting change in categorical data: Mining contrast sets. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 302–306. ACM.
- [3] Buckeridge, D. L., H. Burkom, M. Campbell, W. R. Hogan, and A. W. Moore (2005). Algorithms for rapid outbreak detection: a research synthesis. *Journal of Biomedical Informatics* 38(2), 99–113.

- [4] Cooper, G. F., D. H. Dash, J. D. Levander, W.-K. Wong, W. R. Hogan, and M. M. Wagner (2004a). Bayesian biosurveillance of disease outbreaks. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pp. 94–103. AUAI Press.
- [5] Cooper, G. F., D. H. Dash, J. D. Levander, W.-K. Wong, W. R. Hogan, and M. M. Wagner (2004b). Bayesian biosurveillance of disease outbreaks. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, UAI '04, Arlington, Virginia, United States, pp. 94–103. AUAI Press.
- [6] De Lathauwer, L., B. De Moor, and J. Vandewalle (2000). A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications* 21(4), 1253–1278.
- [7] Dong, G. and J. Li (1999). Efficient mining of emerging patterns: Discovering trends and differences. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 43–52. ACM.
- [8] Dong, Y., Y. Li, and M. Lai (2010). Structural damage detection using empirical-mode decomposition and vector autoregressive moving average model. *Soil Dynamics and Earthquake Engineering* 30(3), 133 – 145.
- [9] Enders, W. and T. Sandler (1993). The effectiveness of antiterrorism policies: A vector-autoregression-intervention analysis. *American Political Science Review* 87(4), 829–844.
- [10] Fanaee-T, H. and J. Gama (2013). Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, 1–15.
- [11] Fawcett, T. and F. Provost (1999). Activity monitoring: Noticing interesting changes in behavior. In *In Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 53–62.
- [12] Gesteland, P. H., R. M. Gardner, F.-C. Tsui, J. U. Espino, R. T. Rolfs, B. C. James, W. W. Chapman, A. W. Moore, and M. M. Wagner (2003). Automated syndromic surveillance for the 2002 winter olympics. *Journal of the American Medical Informatics Association* 10(6), 547–554.
- [13] Gursky, E., T. V. Inglesby, and T. O’toole (2003). Anthrax 2001: observations on the medical and public health response. *Biosecurity and Bioterrorism: Biodefense Strategy, Practice, and Science* 1(2), 97–110.
- [14] Hanely, J. and B. McNeil (1982). The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology* 143(1), 29–36.
- [15] Henning, K. J. (2004). Overview of syndromic surveillance. what is syndromic surveillance. *MMWR Morb Mortal Wkly Rep* 53(Suppl), 5–11.
- [16] Hutwagner, M. L., M. W. Thompson, G. M. Seeman, and T. Treadwell (2003). The bioterrorism preparedness and response early aberration reporting system (ears). *Journal of Urban Health* 80(1), i89–i96.
- [17] Jiang, X. and G. F. Cooper (2010). A bayesian spatio-temporal method for disease outbreak detection. *Journal of the American Medical Informatics Association* 17(4), 462–471.
- [18] Kaufman, Z., E. Cohen, T. Peled-Leviatan, C. Lavi, G. Aharonowitz, R. Dichtiar, M. Bromberg, O. Havkin, Y. Shalev, R. Marom, et al. (2005). Using data on an influenza b outbreak to evaluate a syndromic surveillance system-israel, june 2004. *MMWR (CDC)*(54 (Suppl)), 191.
- [19] Klema, V. and A. Laub (1980). The singular value decomposition: Its computation and some applications. *Automatic Control, IEEE Transactions on* 25(2), 164–176.
- [20] Kolda, T. G. and B. W. Bader (2009). Tensor decompositions and applications. *SIAM review* 51(3), 455–500.



- [21] Ku, W., R. H. Storer, and C. Georgakis (1995). Disturbance detection and isolation by dynamic principal component analysis. *Chemometrics and Intelligent Laboratory Systems* 30(1), 179 – 196. InCINC '94.
- [22] Kulldorff, M. (1997). A spatial scan statistic. *Communications in Statistics-Theory and methods* 26(6), 1481–1496.
- [23] Kulldorff, M. (1999). Spatial scan statistics: models, calculations, and applications. In *Scan statistics and applications*, pp. 303–322. Springer.
- [24] Kulldorff, M., W. Athas, E. Feurer, B. Miller, and C. Key (1998). Evaluating cluster alarms: a space-time scan statistic and brain cancer in los alamos, new mexico. *American journal of public health* 88(9), 1377–1380.
- [25] Kulldorff, M., F. Mostashari, L. Duczmal, W. Katherine Yih, K. Kleinman, and R. Platt (2007). Multivariate scan statistics for disease surveillance. *Statistics in Medicine* 26(8), 1824–1833.
- [26] Kulldorff, M. and N. Nagarwalla (1995). Spatial disease clusters: detection and inference. *Statistics in medicine* 14(8), 799–810.
- [27] Kurucz, M., A. A. Benczúr, and K. Csalogány (2007). Methods for large scale svd with missing values. In *Proceedings of KDD Cup and Workshop*, Volume 12, pp. 31–38. Citeseer.
- [28] Le Strat, Y. and F. Carrat (1999). Monitoring epidemiologic surveillance data using hidden markov models. *Statistics in medicine* 18(24), 3463–3478.
- [29] Lotze, T., G. Shmueli, and I. Yahav (2007). Simulating multivariate syndromic time series and outbreak signatures. *Robert H. Smith School Research Paper No. RHS-06-054*.
- [30] MacGregor, J. and T. Kourti (1995). Statistical process control of multivariate processes. *Control Engineering Practice* 3(3), 403–414.
- [31] Moore, A. W., B. Anderson, K. Das, and W.-K. Wong (2006). Combining multiple signals for biosurveillance. *Handbook of biosurveillance*, 235.
- [32] Moskvina, V. and A. Zhigljavsky (2003). An algorithm based on singular spectrum analysis for change-point detection. *Communications in Statistics-Simulation and Computation* 32(2), 319–352.
- [33] Neill, D.B. (2006). *Detection of spatial and spatio-temporal clusters*. Ph. D. thesis, Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA.
- [34] Que, J. and F.-C. Tsui. A multi-level spatial clustering algorithm for detection of disease outbreaks. *AMIA Annual Symposium proceedings AMIA Symposium AMIA Symposium 2008*, 611–615.
- [35] Rath, T. M., M. Carreras, and P. Sebastiani (2003). Automated detection of influenza epidemics with hidden markov models. In *Advances in Intelligent data analysis V*, pp. 521–532. Springer.
- [36] Reis, B. Y. and K. D. Mandl (2003). Time series modeling for syndromic surveillance. *BMC Medical Informatics and Decision Making* 3(1), 2.
- [37] Serfling, R. E. (1963). Methods for current statistical analysis of excess pneumonia-influenza deaths. *Public health reports* 78(6), 494.
- [38] Shmueli, G. and H. Burkom (2010). Statistical challenges facing early outbreak detection in biosurveillance. *Technometrics* 52(1).
- [39] Shmueli, G. and S. Fienberg (2006). Current and potential statistical methods for monitoring multiple data streams for biosurveillance. In A. Wilson, G. Wilson, and D. Olwell (Eds.), *Statistical Methods in Counterterrorism*, pp. 109–140. Springer New York.
- [40] Siegrist, D. and J. Pavlin (2004). Bio-alert biosurveillance detection algorithm evaluation. *MMWR Morb Mortal Wkly Rep* 53(Suppl), 152–158.

- [41] Simmons, K. M. and D. Sutter (2009). False alarms, tornado warnings, and tornado casualties. *Weather, Climate, and Society* 1(1), 38–53.
- [42] Sun, J., D. Tao, S. Papadimitriou, P. S. Yu, and C. Faloutsos (2008a, October). Incremental tensor analysis: Theory and applications. *ACM Trans. Knowl. Discov. Data* 2(3), 11:1–11:37.
- [43] Sun, J., D. Tao, S. Papadimitriou, P. S. Yu, and C. Faloutsos (2008b). Incremental tensor analysis: Theory and applications. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 2(3), 11.
- [44] Wagner, M. M., F. Chiang Tsui, J. U. Espino, V. M. Dato, D. F. Sittig, R. A. Caruana, L. F. Mcginnis, D. W. Deerfield, M. J. Druzdzal, and D. B. Fridsma (2001). The emerging science of very early detection of disease outbreaks. In *J Pub Health Manag Pract.* 7, pp. 51–59. Addison Wesley.
- [45] Weng-Keen W, Moore, A. and M. Cooper G., Wagner (2013, March). Wsare datasets. <http://www.autonlab.org/autonweb/datasets/15959.html>.
- [46] Williamson, G. D. and G. W. Hudson (1999). A monitoring system for detecting aberrations in public health surveillance reports. *Statistics in Medicine* 18(23), 3283–3298.
- [47] Wong, W.-K., A. Moore, G. Cooper, and M. Wagner (2003, August). Bayesian network anomaly pattern detection for disease outbreaks. In T. Fawcett and N. Mishra (Eds.), *Proceedings of the Twentieth International Conference on Machine Learning*, Menlo Park, California, pp. 808–815. AAAI Press.
- [48] Wong, W.-K., A. Moore, G. Cooper, and M. Wagner (2005, December). What’s strange about recent events (wsare): An algorithm for the early detection of disease outbreaks. *J. Mach. Learn. Res.* 6, 1961–1998.
- [49] Ye, N., Q. Chen, S. M. Emran, and S. Vilbert (2000). Hotelling t<sub>2</sub> multivariate profiling for anomaly detection. *Proc. 1st IEEE SMC Inform. Assurance and Security Workshop*.
- [50] Zhang, J., F.-C. Tsui, M. M. Wagner, and W. R. Hogan (2003). Detection of outbreaks from time series data using wavelet transform. In *AMIA Annual Symposium Proceedings*, Volume 2003, pp. 748. American Medical Informatics Association.

Table 4: Number of False Alarms per year for 100 datasets averaged for 231 p-values (0.020,0.021,..., 0.250)

dataset	WS2.0	WS2.5	WS3.0	EigenEv	dataset	WS2.0	WS2.5	WS3.0	EigenEv
1	70.013	51.8571	57.8485	<b>37.8398</b>	51	54.9827	36.1602	41.013	<b>24.7835</b>
2	0	0	0	0	52	89.7922	<b>51.29</b>	59.9481	51.697
3	<b>0.4416</b>	2.2251	1.8182	1.4545	53	89.2078	62.1991	61.2727	<b>46.3074</b>
4	70.4719	54.3896	53.5368	<b>30.71</b>	54	49.0952	31.7403	33.4762	<b>10.4805</b>
5	69.5974	40.5108	39.5758	<b>29.0823</b>	55	47.4156	42.1991	48.3896	<b>29.2381</b>
6	93.7792	59.0779	62.0346	<b>37.7056</b>	56	108.3766	73.5108	71.3333	<b>40.5498</b>
7	72.5628	50.5584	62.1082	<b>35.3203</b>	57	59.6017	36.7489	38.3117	<b>20.0866</b>
8	89.5628	71.039	79.2771	<b>50.0303</b>	58	25.1299	20.4589	22.1948	<b>13.7706</b>
9	26.3506	17.684	18.4632	<b>8.2468</b>	59	90.4372	41.8658	41.6797	<b>40.0779</b>
10	70.0563	53.8831	49.5584	<b>34.7532</b>	60	79.0996	51.2208	52.3074	<b>50.7013</b>
11	89.6104	63.2251	53.2771	<b>48.3636</b>	61	70.4329	44.29	50.4286	<b>29.316</b>
12	89.3896	71.3333	75.1342	<b>48.4805</b>	62	31.5455	19.1645	19.7879	<b>9.342</b>
13	82.8442	51.0996	49.0346	<b>40.0346</b>	63	59.3463	42.632	45.2511	<b>33.2597</b>
14	35.7013	25.2597	25.4762	<b>17.632</b>	64	72.6407	48.4892	51.9221	<b>37.29</b>
15	80.7619	44.6926	43.6407	<b>34.7879</b>	65	29.2251	19.684	20.2857	<b>9.2684</b>
16	58.8745	42.7056	42.5195	<b>25.7273</b>	66	3.2771	2.3723	3.987	<b>0.8312</b>
17	5.5714	4.2208	6.6753	<b>1.3939</b>	67	39.8398	22.6537	22.9351	<b>10.2468</b>
18	62.0433	42.0606	44.0346	<b>16.2338</b>	68	30.5844	13.4286	15.8052	<b>7.4286</b>
19	55.1472	38.7532	45.6017	<b>32.9567</b>	69	9.0476	8.0087	10.5931	<b>2.9004</b>
20	31.8225	31.3896	30.4675	<b>17.3463</b>	70	1.0952	<b>0</b>	<b>0</b>	<b>0</b>
21	17.3074	9.9221	11.8831	<b>8.3117</b>	71	7.961	7.7489	9.5281	<b>2.7186</b>
22	42.4502	22.3506	20.9091	<b>11.1775</b>	72	9.974	9.645	7.7835	<b>2.9697</b>
23	10.9913	7.2251	8.6667	<b>0.9524</b>	73	35.7749	28.3939	27.0476	<b>16.6667</b>
24	29.4199	20.2338	26.2035	<b>10.1255</b>	74	18.5801	13.7706	13.0563	<b>6.3853</b>
25	35.1905	25.3377	22.7706	<b>14.5455</b>	75	78.0433	44.7316	56.9134	<b>35.2424</b>
26	84.2597	54.7229	57.4589	<b>50.0909</b>	76	73.1082	48.8095	<b>48.6753</b>	52.2165
27	9.987	9.8052	11.2641	<b>4.4372</b>	77	8.8615	8.2944	8.3983	<b>0.2381</b>
28	20.3896	16.4632	14.7316	<b>3.1732</b>	78	29.5238	21.5108	12.5455	<b>10.697</b>
29	0	0	0	0	79	76.4675	53.1732	55.961	<b>50.9437</b>
30	93.2468	65.0736	69.4156	<b>48.4199</b>	80	48.5152	23.4459	23.5325	<b>15.3074</b>
31	47.8701	35.5108	36.8398	<b>32.8528</b>	81	<b>4.2338</b>	10.303	9.1645	4.5801
32	74.7013	60.7835	70.7576	<b>40.3939</b>	82	38.316	22.0779	21.9481	<b>10.3377</b>
33	<b>0</b>	<b>0</b>	0.6537	<b>0</b>	83	85.9004	57.8052	57.1775	<b>40.6797</b>
34	71.7229	40.6623	40.9437	<b>20.0866</b>	84	8.3074	4.7965	4.5152	<b>0.7749</b>
35	88.7706	54.7489	<b>48.5152</b>	52.1472	85	20.303	14.8095	12.4978	<b>3.1255</b>
36	61.2468	50.4719	40.6407	<b>39.8268</b>	86	77.6061	42.0996	49.8745	<b>35.7532</b>
37	31.0866	15.4719	<b>12.961</b>	16.8918	87	22.5671	18.0173	22.4459	<b>8.658</b>
38	31.6104	23.2771	27.3247	<b>18.7662</b>	88	8.3723	6.5238	7.0303	<b>1.316</b>
39	95.3203	65.1385	73.9913	<b>46.8831</b>	89	57.5108	40.0606	51.1212	<b>21.7229</b>
40	33.3939	18.9307	23.9177	<b>15.8874</b>	90	67.71	46.0433	58.1385	<b>24.2381</b>
41	65.0909	39.8918	39.2208	<b>31.29</b>	91	67.7749	48.6797	43.1169	<b>20.7056</b>
42	77.7359	43.7273	49.8528	<b>24</b>	92	68.316	55.5541	56.2035	<b>34.9221</b>
43	30.9264	20.4719	20.5887	<b>10.4762</b>	93	66.987	48.7013	47.1039	<b>28.5022</b>
44	63.0303	46.039	48.8918	<b>36.4416</b>	94	28.697	24.1688	27.7056	<b>9.7576</b>
45	81.7879	47.7792	49.3506	<b>45.2771</b>	95	69.4632	48.1169	47.2944	<b>33.2814</b>
46	84.7749	65.7662	67.8355	<b>41.4589</b>	96	34.0216	16.8528	18.2121	<b>12.4502</b>
47	8.3377	11.1688	9.684	<b>0.5108</b>	97	4.0996	6.1818	7.5498	<b>1.619</b>
48	43.6234	19.7879	29.7489	<b>15.4242</b>	98	34.0476	18.4372	15.0087	<b>14.7403</b>
49	26.5195	<b>14.1861</b>	18.1169	15.4286	99	56.0909	30.987	42.8485	<b>22.7186</b>
50	33.4329	31.6797	35.3117	<b>16.5671</b>	100	66.7922	46.4242	50.5844	<b>28.9394</b>

Table 5: Detection Delay (in days) for 100 datasets averaged for 231 p-values (0.020,0.021,..., 0.250)

dataset	WS2.0	WS2.5	WS3.0	EigenEv	dataset	WS2.0	WS2.5	WS3.0	EigenEv
1	<b>1.3463</b>	2	<b>1.3463</b>	2	51	<b>1</b>	<b>1</b>	<b>1</b>	3.5325
2	<b>1.0433</b>	1	1	1.2987	52	<b>1</b>	<b>1</b>	<b>1</b>	2.0693
3	<b>2</b>	<b>2</b>	<b>2</b>	12.9394	53	<b>3.7835</b>	8.7532	7.7706	4.8312
4	1	1	1	1	54	1	1	1	1
5	<b>1</b>	<b>1</b>	<b>1</b>	2.7273	55	1	1	1	1
6	<b>4.7056</b>	6.2035	5.2511	6.0346	56	1	1	1	1
7	<b>1</b>	<b>1</b>	<b>1</b>	1.5065	57	1	1	1	1
8	1.2251	<b>1</b>	<b>1</b>	1.6104	58	1	1	1	1
9	1.5628	1.2771	1.1688	<b>1</b>	59	5.0909	4.4892	6.1212	11.0606
10	<b>1</b>	9.1905	<b>1</b>	<b>1</b>	60	3.039	2	<b>1.5628</b>	5.4286
11	<b>1</b>	<b>1</b>	<b>1</b>	1.5974	61	<b>1.9957</b>	2	2	2
12	1	1	1	1	62	1	1	1	1
13	4.2078	<b>1</b>	<b>1</b>	7.8139	63	<b>1</b>	2	<b>1</b>	<b>1</b>
14	1	1	1	1	64	4.5065	13.9913	<b>1.7316</b>	2.026
15	2.3593	<b>1</b>	<b>1</b>	1.7316	65	<b>4.0779</b>	7.6364	10.4156	8.4502
16	1.039	<b>1</b>	<b>1</b>	1.0996	66	2	<b>1.3463</b>	2	1.5758
17	<b>1</b>	<b>1</b>	<b>1</b>	1.5065	67	2.8095	<b>1</b>	<b>1</b>	1.3506
18	<b>1</b>	<b>1</b>	<b>1</b>	1.026	68	1	1	1	1
19	1.3074	<b>1</b>	1.2338	2	69	<b>1</b>	<b>1</b>	<b>1</b>	3.8139
20	1	1	1	1	70	<b>1</b>	<b>1</b>	<b>1</b>	2.5931
21	1	1	1	1	71	1	1	1	1
22	1	1	1	1	72	6.5368	<b>3</b>	<b>3</b>	11.5455
23	1	1	1	1	73	<b>4</b>	7.5758	6.3463	5.3333
24	1	1	1	1	74	1.2597	<b>1</b>	<b>1</b>	<b>1</b>
25	1	1	1	1	75	<b>1</b>	2.1255	<b>1</b>	<b>1</b>
26	2.5541	2	<b>1.2035</b>	5.0649	76	<b>1.2121</b>	2	1	1.2251
27	1.4329	<b>1</b>	<b>1</b>	<b>1</b>	77	1.9091	<b>1</b>	<b>1</b>	2.2468
28	1	1	1	1	78	10.0779	<b>7.4632</b>	13.7403	11.2294
29	1	1	1	1	79	<b>4.2554</b>	8.1039	4.961	8.7532
30	8.6883	2.0823	<b>2</b>	4.2338	80	1	1	1	1
31	1	1	1	1	81	1	1	1	1
32	5.4242	1.2597	<b>1</b>	2.3506	82	1.4502	<b>1</b>	<b>1</b>	<b>1</b>
33	1	1	1	1	83	6.6753	<b>4.5022</b>	7.355	5.6147
34	1	1	1	1	84	1.3463	<b>1</b>	<b>1</b>	<b>1</b>
35	3.6883	<b>1.329</b>	1.8788	5.6623	85	1	1	1	1
36	<b>2.2597</b>	6.4935	7.1126	10.1082	86	<b>1.7835</b>	2	2	1.974
37	1	1	1	1	87	2.9134	2.645	1	4.0823
38	1	1	1	1	88	<b>1</b>	<b>1</b>	<b>1</b>	3.0823
39	5.7186	6.4242	10.0649	10.7273	89	1	1	1	1
40	1.0476	<b>1</b>	<b>1</b>	<b>1</b>	90	1	1	1	1
41	1	1	1	1	91	2.1732	1.0866	<b>1</b>	7.1602
42	1.5628	<b>1</b>	<b>1</b>	<b>1</b>	92	6.8095	<b>3.4762</b>	5.1429	11.1602
43	<b>1</b>	10.0779	<b>1</b>	<b>1</b>	93	1.5628	1.3463	<b>1</b>	2.329
44	2.329	<b>1</b>	<b>1</b>	1.961	94	1.2468	1.329	<b>1.1299</b>	1.9827
45	6.3074	2.3853	2.039	<b>1</b>	95	1.4502	<b>1</b>	<b>1</b>	<b>1</b>
46	4.5281	<b>1</b>	<b>1</b>	6.8831	96	1	1	1	1
47	<b>7.3506</b>	10.2944	7.9913	12.9957	97	1	1	1	1
48	1.2727	<b>1</b>	<b>1</b>	1.3593	98	3.645	<b>1</b>	<b>1</b>	1.3377
49	4.8268	3.3463	2.3463	<b>1</b>	99	1	1	1	1
50	1	1	1	1	100	<b>1</b>	<b>1</b>	<b>1</b>	1.9567