Towards efficient prospective detection of multiple spatio-temporal clusters

Bráulio Veloso[1], **Andréa Iabrudi**[1], **Thais Correa**[2]

[1] Computer Science Department

[2]Statistics Department
Universidade Federal de Ouro Preto (UFOP) – Ouro Preto, MG – Brazil

{andrea.iabrudi,thaiscorrea}@iceb.ufop.br, brauliocic091@gmail.com

Abstract. *In this paper we propose a novel technique to efficiently detect multiple emergent clusters in a space-time point process. Emergent cluster detection in large datasets is a ubiquitous task in any application area where fast response is crucial, like epidemic surveillance, criminology or social networks behavior changing. Although different authors investigate aspects of efficient spatio-temporal cluster detection, they handle either multiple or prospective detection of spatio-temporal clusters. Our work concomitantly presents a solution for both aspects: prospective and multiple cluster efficient detection in space and time. Our results with synthetic data are very encouraging, since with a wide range of parameters, we are able to detect multiple clusters in about 90% of the scenarios with very low type I and II errors (less than 2%), without increasing delay time.*

1. Introduction

This work presents a new method for accurate and computationally efficient prospective detection of multiple clusters in space-time event databases, suitable for intensive generating processes. A spatio-temporal cluster is an aggregate of points that are grouped together in space and time with an abnormally high incidence, which has a low probability to have occurred by chance alone. A process that detects such a cluster at earlier stage – an emergent or live cluster – is called surveillance system [Höhle 2007]. Surveillance system development is a ubiquitous task in any application area where fast response is crucial. These application areas include public health and safety surveillance, real life event detection from social network data, traffic control, among others.

Spatio-temporal data is increasingly available as geo-tagged procedures are more popular [Richardson 2013]. Geographic information system community are actively proposing methods to tackle with many different issues [Oliveira and Baptista 2012]: storage, information recovery, ontologies and visualization methods specific for spatio-temporal data. In particular, specialized clustering is a promising and important area for the GIScience and KDD communities [Bogorny and Shekhar 2010, Goodchild 2010]. [**?**] classify spatio-temporal types in: ST Events, Geo-Referenced Variables, Moving Objects and Trajectories. Moving Objects and Trajectories are recent in Geographic Information [Gudmundsson et al. 2012], while Geo-Referenced Variables are usual in Data Mining applications [Birant and Kut 2007] and ST Event, in Computational Statistics [Kulldorff 2001]. Our method classifies as an early distance-based ST event clustering detection procedure.

Epidemiology is traditionally a proficuous area for surveillance systems, as disease outbreak detection is a crucial task, requiring very fast reaction from public agents. Through the years, many different methods have been proposed [Marshall et al. 2007, Tango et al. 2011], investigating alternative point process hypothesis, various cluster shapes, and interaction of Spatio, Spatio-Temporal (ST) and Non-ST data. Usually, the method's quality is measured by the Type I (false clusters identification) and Type II (no detection of true clusters) errors and, for prospective detection, the delay time (elapsed time between the cluster start and its actual identification). There is no single winner method that applies for all situations and applications, and generally the underlying hypothesis are very different and sometimes restrictive. Besides that, recent increase in data availability strengthenes the need of computationally efficient approaches.

In our work, we identify whether there are one or more anomalous concentrations of point events in the very early stage. The underlying assumption is that the points are generated by a Poisson process, with rates that may vary both in space and time. The specific distribution parameters are directly estimated from the data, and may be heterogeneous and non stationary, guaranteeing broad applicability. Furthermore, there is explicit control of Type I error. Computational cost is controlled by considering only cylindrical clusters and using simple likelihood statistics to identify unexpected concentrations, as in [Li et al. 2011].

In a previous work [Veloso et al. 2012], we showed that our method is suitable to handle large volumes of spatio-temporal data to discover actual events from social network message. This work is a step towards bringing together Statistical Computing and GIScience applications, by allowing existence of multiple clusters, which is likely when the generating process is intense. Detection of multiple clusters has been investigated by spatial and spatial-temporal retrospective cluster detection community. In [Zhang et al. 2010], a sequential version of the spatial scan statistic procedure is adjusted for the presence of other clusters in the study region, by sequential deletion of the previously detected clusters, much like as our approach. In [Li et al. 2011], the authors consider the existence of multiple clusters directly by the alternative hypothesis and show better power in terms of both rejecting the null hypothesis and accurately detecting the coexisting clusters. Both methods are not suitable to emergent detection and have significant computational cost. Adopting a graph-based strategy, [Demattei and Cucala 2010] identify clusters by linking the events closest than a given distance and thus defining a graph associated to the point process. The set of possible clusters is then restricted to windows including the connected components of the graph. This allow detection of multiple, arbitrary shaped clusters with relatively efficiency. There is no extension for prospective detection, as far as we know.

Our main contribution in this paper is to extend the method in [Assunção and Correa 2009] to identify multiple clusters, using a sequential strategy of deleting some events of already identified clusters. The proposed method is evaluated for a wide range of synthetic scenarios, simulating rare to intensive processes, where two spatially separated clusters are inserted. In this controlled setup, our results are very promising, with more than 93% of correct detections. We first present the original method, followed by our extension to couple with multiple clusters. The metrics used to evaluate the method are detailed and results are then presented and discussed.

Final remarks and possible future directions close the article.

2. Method

We extended the method proposed by [Assunção and Correa 2009] for detection of multiple space-time clusters. In section 2.1, the original method, designed for an unique space-time cluster detection is briefly presented. We state our extension in section 2.2, proposing the algorithms for it.

2.1. Review: on line detection of an unique space-time cluster

Consider a point process observed in a three dimensional area $A \times (0, T]$ where A represents the space and $(0, T]$ is the time. The original method looks for a live space-time cluster with cylindric shape. The radius ρ of the circular base must be specified by the user. The method consist on monitoring a simple statistic that doesn't depend on the marginals space and time intensities of events. When this statistic exceeds a threshold the method rings an alarm and the detected cluster is identified.

Events are sequentially observed at times $t_1, t_2, \ldots$ and they are processed as soon as they happen. The spatial coordinates for an event observed at time t_i are (x_i, y_i). Let $C_{k,n} \in A \times (0, T]$ be a cylinder with circular base spatially centered at (x_k, y_k). The height of $C_{k,n}$ is given by $t_n - t_k$, where t_n is the time of the last observed event. Note that, considering t_n as the current time, $C_{k,n}$ is a live cylinder, since it reaches t_n. Let $N(C_{k,n})$ be the number of events inside the cylinder $C_{k,n}$, we assume that it follows a Poisson distribution, $N(C_{k,n}) \sim \text{Poisson}(\mu(C_{k,n}))$. The Poisson distribution is broadly used in methods of cluster detection due to its suitability for modeling count data. If there is no cluster, no space-time dependency exists, implying that space-time event intensity $\lambda(x, y, t)$ is separable and may be written as the product of the space and time marginal intensities:

$$\lambda(x, y, t) = \mu\, \lambda_s(x, y)\, \lambda_t(t), \qquad \mu = \int_A \int_{(0,T]} \lambda(x, y, t)\, dt\, dx\, dy.$$

However, if a cluster $C_{k,n}$ emerges at time t_k, dependency change in distribution is captured by a constant $\varepsilon > 0$ such that

$$\lambda(x, y, t) = \mu\, \lambda_s(x, y)\, \lambda_t(t)\, (1 + \varepsilon I_{C_{k,n}}(x, y, t))\,,$$

where $I_{C_{k,n}}$ is an indicator function for $(x, y, t) \in C_{k,n}$ and ε represents the intensity increase inside the cluster. The excess ε is a method parameter specified by the user.

The test statistic compares the null hypothesis of no cluster existence against a localized cylindrical cluster alternative. Let L_∞ be the likelihood of the spacetime Poisson process when there is no cluster and let L_k be the likelihood of this same process when there is a cluster $C_{k,n}$, both for n observed events. The test statistic is the sum of the likelihood ratio over all possibilities for the cylinder $C_{k,n}$:

$$R_n = \sum_{k=1}^{n} \frac{L_k}{L_\infty} = \sum_{k=1}^{n} \Lambda_{k,n} = \sum_{k=1}^{n} (1 + \varepsilon)^{N(C_{k,n})}\, \exp(-\varepsilon\, \mu(C_{k,n}))\,.$$

The method uses a non parametric estimate for the mean $\mu(C_{k,n})$, as in practice it is unknown. Assuming that $\lambda(x, y, t)$ is separable, our method estimates this quantity by:

$$\hat{\mu}(C_{k,n}) = \frac{N(\ B(k,\rho) \times (0,t_n]\ )\ N(\ \mathcal{A} \times (t_k,t_n]\ )}{n}\ ,$$

where $N(\ B_{k,\rho} \times (0,t_n]\ )$ is the number of events inside the circular base of cylinder $C_{k,n}$ irrespective of time, $N(\mathcal{A} \times (t_k,t_n])$ is the number of event between time t_k and t_n irrespective of space, and n is the total number of events.

When the test statistic exceeds a threshold A, the method rings an alarm indicating there is evidence of a live cluster. As the test statistic is a sum over all possible live clusters, the estimate of the detected live cluster is the one with largest contribution to the test statistic. That is, if $R_n > A$, then $\Lambda_{k*,n} = \max\{\Lambda_{k,n}, 1 \leq k \leq n\}$ and the estimated cluster is $C_{k*,n}$.

Assume that there is a cluster starting at time t_k. If the test statistic exceeds the threshold A at some time $t < t_k$, then it is a false alarm. Otherwise, if threshold A is exceeded at time $t > t_k$, it is a motivated alarm. Surveillance systems must address the trade-off between fast detection and low false alarm rate. In our method, this is accomplished by setting the threshold A to be equal to the desired value of the Average Run Length (ARL), the expected number of events before a false alarm. It ensures that, on average, the user will wait at least A events before a false alarm, expressing user tolerance to wrong alerts.

2.2. Our extension for simultaneous space-time clusters

Now suppose we are using the method above and the alarm rings. It means there is evidence of a live cluster. In this case, is there evidence of a second live cluster? What about a third live cluster? We answer these questions with our extension for the situation where more than one cluster start at the same time t_k. The radius and the increase in the intensity for all clusters are the same: ρ and ε, respectively. For simplicity, we describe the extension for two clusters, but it can be generalized for any number of simultaneous clusters.

In the original method with a threshold A, consider there is an alarm at time t_n. The estimated cluster is $C_{k*,n}$. Our sequential approach consists in deleting the excess of events inside $C_{k*,n}$ in a random way and reapplying the original method to this new reduced database. The excess of events is the difference between the observed and the expected number of events: $\Delta(C_{k*,n}) = N(C_{k*,n}) - \hat{\mu}_{k*,n}$.

After deletion of the exceeding events, we have a reduced database. We then evaluate the test statistic at the current time t_n for this new reduced database. We will refer to the statistic for the reduced database as R'_n to distinguish from R_n. Since we artificially erased the cluster $C_{k*,n}$, if R'_n is more than a new threshold it is due to a second live cluster, it is hoping one different from $C_{k*,n}$. The estimate for this second cluster is the one with largest contribution to the test statistic R'_n, just as in the original method. The new threshold A' is equal to the old one, except for the events we delete: $A' = A - \Delta(C_{k*,n})$.

Algorithms 1, 2, 3 and 4 show our extension.

Algorithm 1 Simultaneous Spatio-Temporal Clusters Detection

Require: E order by time.

function SIM-STCD(n-array of spatio-temporal events E, radius ρ, increase in the intensity ε, threshold A, number of events n)
  Let C be a cluster set initially empty.
  $\Lambda,\ N,\ \hat{\mu},\ M \Leftarrow STCD(\ E,\ \rho,\ \varepsilon,\ n\ )$
  $R_n \Leftarrow \sum_{j=1}^{n}(\ \Lambda_{j,n}\ )$
  alarm $\Leftarrow (\ R_n > A\ )$
  while alarm **do**
    $(C_{k,n},\ E\star) \Leftarrow findCluster(\ E,\ \Lambda,\ M)$
    $C \Leftarrow C \cup \{(C_{k,n},\ E\star)\}$
    $\Delta = N(C_{k,n}) - \hat{\mu}(C_{k,n})$
    $E \Leftarrow removeExcessEvents(\ E,\ E\star,\ \Delta\ )$
    $A \Leftarrow A - \Delta$
    $n \Leftarrow n - \Delta$
    $\Lambda,\ N,\ \hat{\mu},\ M \Leftarrow STCD(\ E,\ \rho,\ \varepsilon,\ n\ )$
    $R_n \Leftarrow \sum_{j=1}^{n}(\ \Lambda_{k,n}\ )$
    alarm $\Leftarrow (\ R_n > A\ )$
  end while
  return C
end function

Algorithm 2 Spatio-Temporal Cluster Detection

Require: E order by time.

function STCD(n-array of spatio-temporal events E, radius ρ, increase in the intensity ε, id of the last event n)
  Let N be an n-array of number of events inside a cylinder
  Let $\hat{\mu}$ be an n-array of expected number of events inside a cylinder
  Let M be an all-zero $n \times n$-matrix of events neighborhood
  Let Λ be an n-array of parcels
  for i $\leftarrow 1$ **to** n **do**
    for j $\leftarrow 1$ **to** n **do**
      $d \Leftarrow spatialDistance(\ E_i = (x_i, y_i),\ E_j = (x_j, y_j)\ )$
      $M_{i,j} \Leftarrow (\ (d \leqslant \rho) \wedge (i \neq j)\ )$
    end for
  end for
  for k $\leftarrow 1$ **to** n **do**
    $N(C_{k,n}) \Leftarrow \sum_{i=k}^{n}(\ M_{n,i}\ )$
    $\hat{\mu}(C_{k,n}) \Leftarrow \frac{(n-k+1)}{n} \sum_{i=1}^{n}(\ M_{k,i}\ )$
    $\Lambda_{k,n} \Leftarrow (1+\varepsilon)^{N(C_{k,n})}\ \exp(-\varepsilon\, \hat{\mu}(C_{k,n}))$
  end for
  return $\Lambda,\ N,\ \hat{\mu},\ M$
end function

Algorithm 3 function findCluster

function FINDCLUSTER(n-array of spatio-temporal events E, parcel's array Λ, events neighborhood matrix M)
  $\Lambda_{k*,n} \Leftarrow MAX\{\Lambda_{k,n}, 1 \leqslant k \leqslant n\}$
  Then, let $C_{k*,n}$ be the cylinder that define the cluster found beginning in event $k*$ and finishing in time of event n
  Let $E^\star$ be a dataset empty
  $E^\star \Leftarrow E^\star \cup \{E_{k*}\}$
  for i $\leftarrow (k*+1)$ **to** n **do** ▷ note: As data are ordered by time, only the events greater than $k*$ can be inside the cylinder $C_{k*,n}$.
    if event i is a $k*$ neighbor, ie., $M_{i,k*} = 1$
      then $E^\star \Leftarrow E^\star \cup \{E_i\}$
    end if
  end for
  return ($C_{k*,n}$, $E^\star$)
end function

Algorithm 4 procedure removeExcessEvents

procedure REMOVEEXCESSEVENTS(spatio-temporal events dataset E, spatio-temporal events sub-dataset $E^\star$, number of events in excess Δ)
  Let E' be an empty dataset
  for i $\leftarrow 1$ **to** Δ **do**
    Sort an event E_k of $E^\star$, always a different one.
    $E' \Leftarrow E' \cup \{E_k\}$
  end for
  $E \Leftarrow E - E'$
end procedure

3. Evaluation metrics

We defined some metrics to evaluate the method for simulated databases. We first describe these metrics when an unique cluster is present. In this case, we analyzed the percentages of *No Alarm*, *Incorrect Alarm*, and *Correct Alarm*. A *No Alarm* occurs when $R_i < A$ for all $i = 1, \ldots, n$, where n is the total number of events in the database. An *Incorrect Alarm* occurs when $R_i > A$ for some $i = 1, \ldots, n$ and the events set of the estimated cluster has no intersection with the events set of the real one. A *Correct Alarm* happens when $R_i > A$ for some $i = 1, \ldots, n$ and the events set of estimated cluster has any intersection with the events set of real one. For each database we used, we let the time moves forward until a *Correct Alarm* and record the total number of different alarms. If $R_i > A$, $R_{i+1} < A$, $R_{i+2} > A$, then the alarms at times i and $i + 2$ was considered as different alarms. When $R_i > A$, $R_{i+1} > A$ and the estimated clusters at times i and $i + 1$ are the same, we considered the alarms at times i and $i + 1$ as the same alarm. If $R_i > A$, $R_{i+1} > A$ and the estimated clusters at time i and $i + 1$ are not the same, we considered the alarms at times i and $i + 1$ as different alarms. We consider the estimated clusters as the same if the distance between their centers is greater than 2ρ.

The percentages of *No Alarm*, *Incorrect Alarm*, and *Correct Alarm* was calculated in relation to the total number of different alarms in all databases. To enable the calculation of these percentages, the total number of different alarms was consider as one for a *No Alarm* database. Figure 1 illustrates these concepts. Situation (a) is the case where the first alarm is a correct one. In (b) there is a period of incorrect alarms before the correct one. We count the number of different incorrect alarms that appear in this period. It is also possible to alternate periods between no alarm and incorrect alarms before reach a correct one. If there isn't any correct alarm, as in situation (c), we count only the number of different incorrect alarms. Finally, situation (d) shows a *No Alarm* case: there is no alarm, neither correct or incorrect.

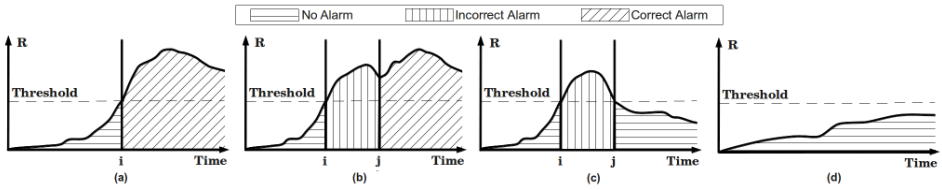

Figure 1. Alarms for some situations with one cluster.

For databases with one cluster, we also record the delay for a *Correct Alarm*, in time unit. This delay is the difference between the time of the the *Correct Alarm* and the real time of the beginning of the cluster.

We now describe the measures used for the situation with two clusters. In this case, an alarm can be *Single* or *Double*. It is a *Single Alarm* when $R_i > A$ and $R'_i < A'$; it is a *Double Alarm* when $R_i > A$ and $R'_i > A'$. Figure 2 illustrates the possibilities for the two clusters situation. Initially it has no alarm (0). If only the first threshold is exceeded (1), then it's a *Single Alarm*. If the first and second thresholds are exceeded (2), then it's a *Double Alarm*. From (1) there is three possibilities. The first one: the estimated cluster changes (3), it's a new *Single Alarm*. The second one: the first threshold is not exceeded anymore (4), then it returns to the case of no alarm. The last one: the second threshold is also exceeded (5), then it's a *Double Alarm*. From (2) there is also three possibilities. If at least one of the estimated clusters change (6), it's a new *Double Alarm*. If the first and second thresholds are not exceeded anymore (7), it returns to the case of no alarm. If only the second threshold is not exceeded anymore (8), then it's a *Single Alarm*.

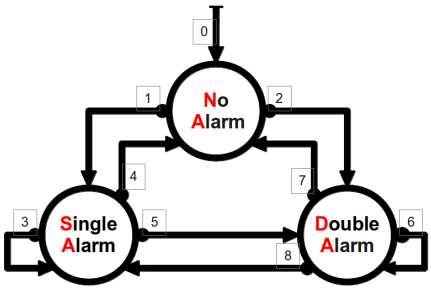

Figure 2. Alarms for the situation with two clusters.

A *Single Alarm* can be *Correct* or *Incorrect*. A *Single Alarm* is *Single Correct Alarm* if the estimated cluster has any intersection with one of the two real clusters, named here $C1$ and $C2$. If the estimated cluster has no intersection with $C1$ and no intersection with $C2$ it is a *Single Incorrect Alarm*. A *Double Alarm* can be *Correct*, *Incorrect* or *Half-Correct*. A *Double Correct Alarm* occurs when one of the estimated clusters has any intersection with $C1$ and the other estimated cluster has any intersection with $C2$. A *Double Incorrect Alarm* occurs the estimated clusters has no intersection with both $C1$ and $C2$. Finally, it is a *Double Half-Correct Alarm* when only one of the estimated clusters has any intersection with $C1$ or $C2$.

For each database with two clusters, we let the time moves forward until a *Double Correct Alarm* and record the total number of different alarms. We analyzed the percentages of *No Alarm*, *Incorrect Alarm*, *Incomplete Alarm*, and *Complete Alarm* in relation to the total number of different alarms in all databases. Again, to enable the calculation of these percentages, the total number of different alarms was consider as one for a *No Alarm* database. Here, *Single Incorrect Alarm* and *Double Incorrect Alarm* were both considered as *Incorrect Alarm*. A *Complete Alarm* is a *Double Correct Alarm*. A *Single Correct Alarm* is an *Incomplete Alarm*. One *Double Half-Correct Alarm* represents $1/2$ *Incorrect Alarm* and $1/2$ *Incomplete Alarm*.

4. Results

In this section we present the results we obtained applying both the original method and our extension to artificial (simulated) data. The original method was applied to databases with an unique cluster and the extension was applied to databases with two simultaneous clusters. We first describe the artificial databases we used in subsection 4.1 and then show the results in the following subsections.

4.1. Simulated databases

In all databases we considered a 10×10-square as the space area and the the time ranging from 0 to 10. We used four different values for the spatial radius $\rho : 0.5, 1.0, 1.5, 2.0$. For the increase in the intensity inside the cluster ε we tried three different values: $1, 3, 10$. All clusters finish at time 10. We also varied the time the cluster emerges. We used $5, 7, 8$ for this initial time. In case of two clusters, time, radius ρ and the excess ε are varied equality for both clusters. The clusters' centers are distant by at least 4ρ, guaranteeing that no cluster candidate intersects both of them simultaneously. In subsections 4.2 and 4.3 the true values for ρ and ε was used as input for these parameters. The threshold for the alarm was always set as the total number of event in the database.

For of each combination of ρ, ε and initial time, we generated 100 databases. Figure 3 presents examples for database's cases with one and two clusters. We show the simulations results in the following subsections. The initial time of the cluster proved not to be significant, and then we show here the average of each measure for the three different values we tried for this time. We also disregard the results for the combinations $\rho = 0.5$, $\varepsilon = 1$ and $\rho = 2.0$, $\varepsilon = 10$, since the method proved to be inadequate in these extremes.

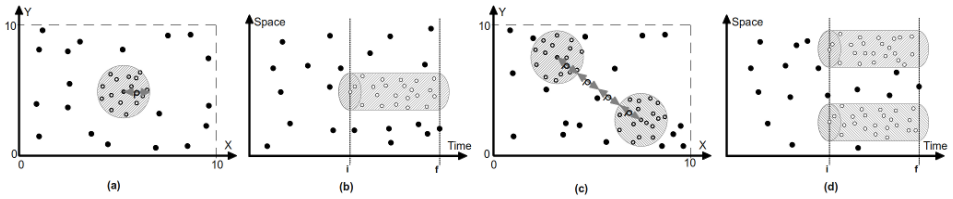

Figure 3. Examples of simulated database. (a) and (b) unique cluster database. (c) and (d) simultaneous clusters database. (a) and (c) space distribution. (b) and (d) space-time distribution.

4.2. Simulation results for an unique cluster

We applied the original method for simulated databases with one cluster. For each database we let the time move forward until a *Correct Alarm* or the final time ($t = 10$). We analyze the percentage of *No Alarm*, *Incorrect Alarm*, and *Correct Alarm*, shown in Figure 4. In this figure the bars represent the percentage of *No Alarm*, *Incorrect Alarm* and *Correct Alarm*, in this order. The segment in each bar is the 95% confidence interval. In all cases the percentage of *No Alarm* and *Incorrect Alarm* are very small.

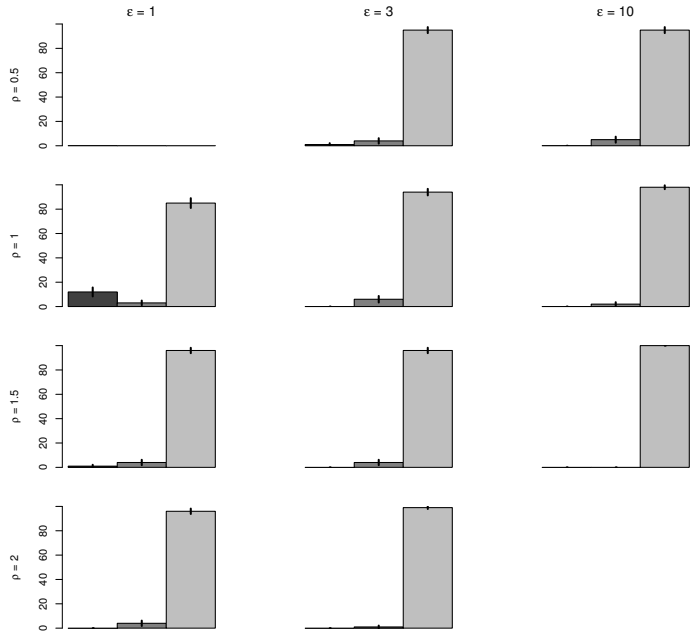

Figure 4. Number of Alarms for an unique cluster. The bars represent the percentage of *No Alarm*, *Incorrect Alarm* and *Correct Alarm*, in this order. The segment in each bar is the 95% confidence interval.

We also measured the delay for a correct alarm, in unit time. On average, it is 0.277, and the 95% confidence interval is $(0.067,\ 0.775)$.

4.3. Simulation results for simultaneous clusters

We applied our extension for simulated databases with two simultaneous clusters, $C1$ and $C2$. For each database we let the time move forward until a *Double Correct Alarm* or the final time ($t = 10$) and count the number of distinct alarms per case into database. The results are shown in Figure 5. In this figure the bars represent the percentage of *No Alarm*, *Incorrect Alarm*, *Incomplete Alarm* and *Complete Alarm*, in this order. The segment in each bar is the 95% confidence interval. The percentage of *No Alarm* and *Incorrect Alarm* are always very small, as in the case of an unique cluster.

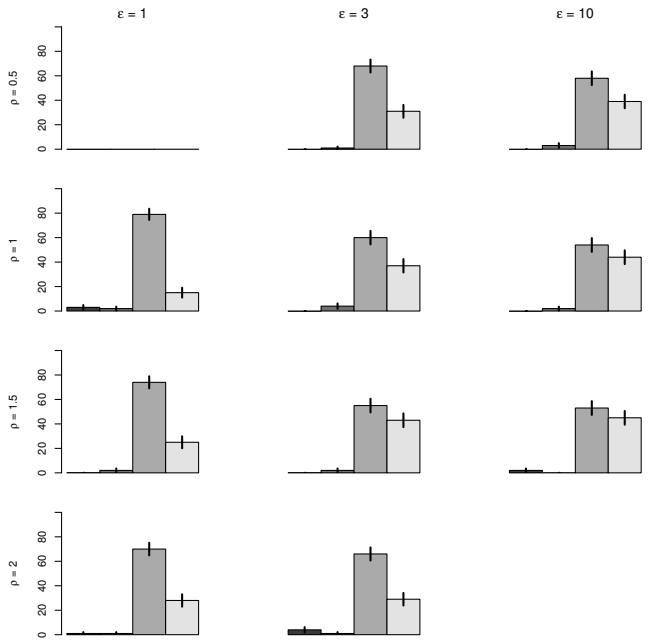

Figure 5. Number of Alarms for two clusters. The bars represent the percentage of *No Alarm*, *Incorrect Alarm*, *Incomplete Alarm* and *Complete Alarm*, in this order. The segment in each bar is the 95% confidence interval.

Figure 6 shows the delay for both the cases with one cluster and two simultaneous clusters. The first bar represents the delay for a *Correct Alarm* when there is an unique cluster (*Delay 1*). The second, third and fourth bars show the delay when there is two simultaneous clusters. These bars represents, in this order, the delay until: the first *Correct Alarm* (*Delay Min*), the detection of cluster $C1$ (*Delay C1*), the detection of cluster $C2$ (*Delay C2*), and a *Double Correct Alarm* (*Delay Double*). The segment in each bar is the 95% confidence band: percentiles 2.5% and 97.5%. We did not found significant difference for the average delay between *Delay 1* and *Delay Min*. It means that, the extension for multiple clusters detects one cluster as fast as the original method. As expected, there is also no significant difference for the average delay between *Delay C1* and *Delay C2*. For all others combinations we found significant difference for the average delay. Note that, on average, *Delay Double* is usually more than *Delay C1* and *Delay C2* and the percentage of *Incomplete Alarm* is always more than the percentage of *Complete*

Alarm. It's means: even before a *Double Correct Alarm*, our extension detects both clusters in different alarms (*Single Correct Alarms*).

Independent of the number of alarms, on average, our extension reached a *Complete Alarm* in 88.2% of cases into database, while 10.6% of cases it only identifies one cluster of two expected (*Incomplete Alarms*) all the time. In 0.2% of cases have only *Incorrect Alarms* and 1% of cases have *No Alarm*.

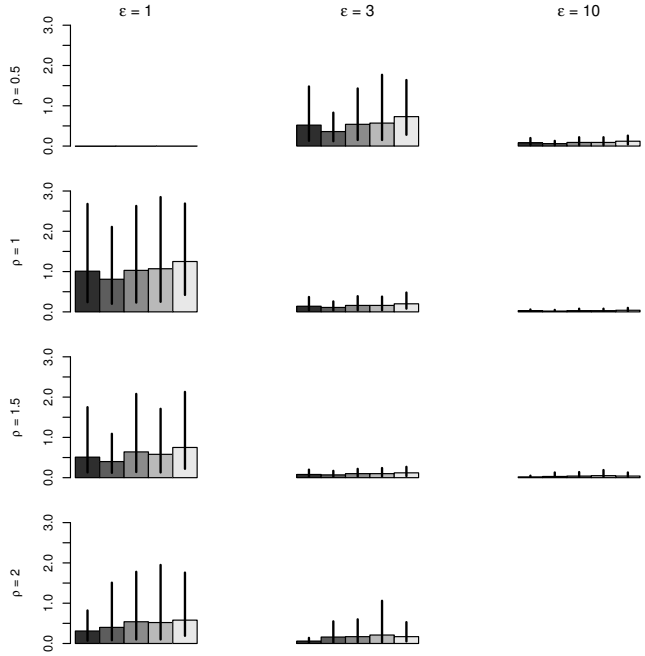

Figure 6. Delays in time unit. The segment in each bar is the 95% confidence band for: *Correct Alarm for one cluster case*, *first Correct Alarm for two cluster case*, *detection of cluster C1*, *detection of cluster C2*, and *Double Correct Alarm*.

5. Final considerations

Simulation results in the previous section show that our extension for multiple cluster is quite satisfactory for spatially separate clusters. It detects one of the multiple clusters as fast as the original method. The percentage of detection for both clusters is around 88%, and the delay is reasonably small. These are initial results, evaluating our method capability of detecting simultaneous clusters with respect to the original one.

Here we took ρ and ε parameters to be exactly their true values on datasets. The impact of changing these parameters was evaluated for the original method in [Assunção and Correa 2009]. The same should be done for the extension for multiple clusters in a future work, as well as the application of the extension to real data. Other future direction is to compare our approach to others, establishing its relative efficiency and effectiveness. Important issues to be considered hereafter are the automatic calibration for these parameters and removing the restriction on the cylindrical shape of the clusters, allowing for arbitrary shaped ones.

References

Assunção, R. and Correa, T. (2009). Surveillance to detect emerging space-time clusters. *Comput. Stat. Data Anal.*, 53(8):2817–2830.

Birant, D. and Kut, A. (2007). ST-DBSCAN: An algorithm for clustering spatial–temporal data. *Data & Knowledge Engineering*, 60(1):208–221.

Bogorny, V. and Shekhar, S. (2010). Spatial and Spatio-temporal Data Mining. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 1217–1217.

Demattei, C. and Cucala, L. (2010). Multiple spatio-temporal cluster detection for case event data: an ordering-based approach. *Communications in Statistics-Theory and Methods*, 40(2):358–372.

Goodchild, M. F. (2010). Twenty years of progress: GIScience in 2010. *Journal of Spatial Information Science*, (1):3–20.

Gudmundsson, J., Laube, P., and Wolle, T. (2012). Computational Movement Analysis. In Kresse, W. and Danko, D. M., editors, *Springer Handbook of Geographic Information*, pages 423–438. Springer Berlin Heidelberg.

Höhle, M. (2007). surveillance: An R package for the monitoring of infectious diseases. *Computational Statistics*, 22:571–582.

Kulldorff, M. (2001). Prospective time periodic geographical disease surveillance using a scan statistic. *Journal of the Royal Statistical Society, Series A*, 164:61–72.

Li, X.-Z., Wang, J.-F., Yang, W.-Z., Li, Z.-J., and Lai, S.-J. (2011). A spatial scan statistic for multiple clusters. *Mathematical biosciences*, 233(2):135–142.

Marshall, J. B., Spitzner, B. D., and Woodall, W. H. (2007). Use of the local Knox statistic for the prospective monitoring of disease occurrences in space and time. *Statistics in Medicine*, 26:1579–1593.

Oliveira, M. G. d. and Baptista, C. d. S. (2012). GeoSTAT: A system for visualization, analysis and clustering of distributed spatiotemporal data. In *Proceedings XIII GEOINFO,*, pages 108–119.

Richardson, D. B. (2013). Real-Time Space–Time Integration in GIScience and Geography. *Annals of the Association of American Geographers*, 103(5):1062–1071.

Tango, T., Takahashi, K., and Kohriyamma, K. (2011). A space-time scan statistic for detecting emerging outbreaks. *Biometrics*, 67:106–115.

Veloso, B. M., Iabrudi, A. I., and Correa, T. R. (2012). Localização em tempo real de acontecimentos através de vigilância espaço-temporal de microblogs. page 12. IX Encontro Nacional de Inteligência Artificial.

Zhang, Z., Assunção, R., and Kulldorff, M. (2010). Spatial scan statistics adjusted for multiple clusters. *Journal of Probability and Statistics*, 2010.