Spatio-temporal Outlier Detection in Precipitation Data

Elizabeth Wu, Wei Liu, and Sanjay Chawla

School of Information Technologies,
The University of Sydney,
Sydney, Australia
{ewu1,weiliu,chawla}@it.usyd.edu.au
http://www.it.usyd.edu.au

Abstract. The detection of outliers from spatio-temporal data is an important task due to the increasing amount of spatio-temporal data available and the need to understand and interpret it. Due to the limitations of current data mining techniques, new techniques to handle this data need to be developed. We propose a spatio-temporal outlier detection algorithm called Outstretch, which discovers the outlier movement patterns of the top-k spatial outliers over several time periods. The top-k spatial outliers are found using the **Exact-Grid Top-k** and **Approx-Grid Top-k** algorithms, which are an extension of algorithms developed by Agarwal et al. [1]. Since they use the Kulldorff spatial scan statistic, they are capable of discovering all outliers, *unaffected by neighbouring regions that may contain missing values*. After generating the outlier sequences, we show one way they can be interpreted, by comparing them to the phases of the El Niño Southern Oscilliation (ENSO) weather phenomenon to provide a meaningful analysis of the results.

Keywords: Spatio-Temporal, Data Mining, Outlier Detection, South America, Precipitation Extremes.

1 Introduction

Spatio-temporal data mining is the discovery of interesting spatial patterns from data over time using data mining techniques on spatially and temporally distributed data. One such pattern is a spatio-temporal outlier.

A spatio-temporal outlier is a spatio-temporal object whose thematic (non-spatial and non-temporal) attributes are significantly different from those of other objects in its spatial and temporal neighbourhoods. An extended discussion of the definition of a spatio-temporal outlier is provided in Section 2.

The interest in utilising spatio-temporal data mining techniques for the discovery of outliers in space and time has been prompted by the increasing amount of spatio-temporal data available and the need to interpret it effectively [2]. The spatio-temporal outliers we discover are of interest because they show particular regions which have precipitation behaviour that is significantly different from nearby regions over some period of time. Understanding the patterns of such

M.M. Gaber et al. (Eds.): Sensor-KDD 2008, LNCS 5840, pp. 115–133, 2010.

behaviour over several years could allow us to find relationships between these regional patterns, and other weather patterns such as the El Niño Southern Oscillation (ENSO). Additionally, with enough historical data, we could examine the historical behaviour of rainfall to try and predict future extreme events, their duration, and their locations.

Another driving force behind the increasing popularity of data mining tools for spatio-temporal outlier detection is the inadequacies of existing methods. Traditional data mining techniques have limitations in the discovery of outliers [3]. Such algorithms can be restrictive as outliers do not fit to a specified pattern or have a usual type of behaviour. In addition, these algorithms are often computationally inefficient since finding all the different behavioural patterns of objects may require more time and effort than that required to find outliers.

The ability to discover an increased number of patterns more accurately could enhance our understanding of many different application areas. This research focuses on the field of Hydrology, where knowledge about the behaviour of unusual precipitation could allow governments and individuals to better prepare for extreme events such as floods. Performing data mining on geographic data forms one part of the process of Geographic Knowledge Discovery (GKD), which is 'the process of extracting information and knowledge from massive geo-referenced databases' [3]. Geographic data mining is the use of computational techniques and tools to discover patterns that are distributed over geographic space and time [4], while taking into consideration data features that are specific to geographic domains [5]. In this study, we conduct our experiments on South American precipitation data provided by the National Oceanic and Atmospheric Administration (NOAA) [6], and as such, we also need to consider the geographical features of the data.

Following this, we compare our results to ENSO using the Southern Oscillation Index (SOI), since it is believed that there is a teleconnection or relationship between ENSO and precipitation. ENSO consists of two phases - El Niño and La Niña. The SOI is a measure of the strength and phase of an El Niño or La Niña event. A prolonged negative SOI is associated with an El Niño event, while a positive SOI is associated with a La Niña event [7]. We obtained the values for the SOI from the NOAA Climate Prediction Center [8]. The calculation of the SOI by the NOAA is provided in greater detail in [9]. In South America, El Niño is associated with increased precipitation over parts of South America, so understanding the relationship between extreme precipitation events and ENSO is an important task.

We make the following contributions in this paper:

- Extend the **Exact-Grid** and the more efficient **Approx-Grid** algorithms from [10] to create the **Exact-Grid Top-k** algorithm and the **Approx-Grid Top-k** algorithms, which find the top-k highest discrepancy regions, rather than only the single highest discrepancy region.
- Develop techniques for handling missing data, and dealing with region selection problems encountered when extending these algorithms to find the top-k regions.

- Extend the above algorithms to use Outstretch to find and store sequences of high discrepancy regions over time into a tree structure.
- We provide the **RecurseNodes** algorithm to allow the extraction of all possible sequences and sub-sequences from the **Outstretch** tree-structure.
- We apply **Outstretch** to South American precipitation data, to show that it is capable of finding outlier sequences from the data set.
- We illustrate one way the data could be analysed by comparing the discrepancy of regions to the SOI.

These contributions focus on assisting the Outstretch algorithm to find the moving paths of the most significant outlier regions over several time periods. This makes analysis of the spatio-temporal nature of historical extreme events possible, and could help to predict the location, duration and timing of future extreme events. To the best of our knowledge, the discovery of such patterns has not previously been investigated.

This paper is organised as follows. Section 2 defines and describes the properties of a spatio-temporal outlier. Section 3 provides an overview of related work. Section 4 describes our method of discovering spatio-temporal outliers. Section 5 outlines the experimental setup and the results we achieve. A discussion of our technique and results, followed by a conclusion and suggestions for further research is given in Section 6.

2 Spatio-temporal Outliers

An outlier can constitute different things in different application settings, and as a result the precise definition of an outlier is difficult to capture [13]. Spatio-temporal outlier detection is an extension of spatial outlier detection. It aims to find spatial outliers, but instead of just looking at a single snapshot in time, it considers the behaviour of these outliers over several time periods. Cheng and Li [11] define a spatio-temporal outlier to be a "spatial-temporal object whose thematic attribute values are significantly different from those of other spatially and temporally referenced objects in its spatial or/and temporal neighbourhoods". Birant and Kut [12] define a spatio-temporal outlier as "an object whose non-spatial attribute value is significantly different from those of other objects in its spatial and temporal neighborhood". From these definitions, Ng [13] notes two possible types of outliers - extreme and non-extreme. If the values under examination follow a normal distribution, then extreme values at the tail of the distribution are classified as extreme value outliers. We focus on this type of outlier as they are indicative of extreme events.

Since a spatio-temporal outlier is a spatio-temporal object, we also need to provide a definition of a spatio-temporal object. Theodoridis *et al.* [14] define a spatio-temporal object as a time-evolving spatial object whose evolution or 'history' is represented by a set of instances (o_id, s_i, t_i), where the spacestamp s_i, is the location of object o_id at timestamp t_i. According to this definition, a two dimensional region is represented by a solid in three-dimensional space. A

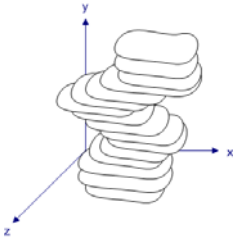
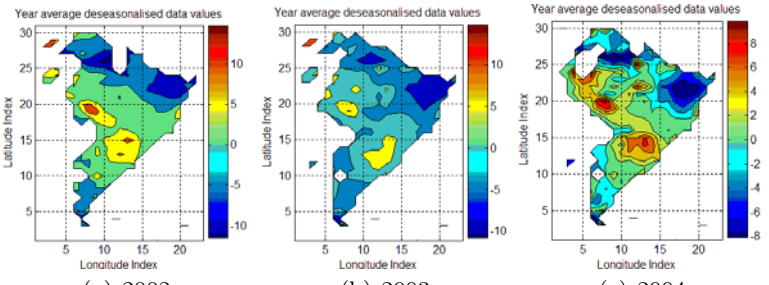


Fig. 1. A moving region



(a) 2002 (b) 2003 (c) 2004

Fig. 2. Deseasonalised Precipitation Data from the South American NOAA Precipitation Dataset

conceptual example of a moving region is shown in Figure 1. A moving region identified from our dataset can be seen in Figure 2 in the grid whose bottom left corner identified as longitude 10 and latitude 10.

3 Related Work

In previous work, Birant and Kut [12] define two spatial objects (S-objects) as temporal neighbors if the "values of these objects are observed in consecutive time units such as consecutive days in the same year or in the same day in consecutive years". However, Birant and Kut regard a spatio-temporal outlier to be a spatial outlier from a single time period that is different from its immediate temporal neighbours. This is essentially a spatial outlier in spatio-temporal data. Since we follow the definition of a spatio-temporal object provided by Theodoridis *et al.* [14], and consider that a spatio-temporal outlier may exist over more than one time period. For example, in our work if there is higher than average precipitation in Peru over the years 1998-2002, then the solid in three dimensional space is an outlier. In experiments conducted by Birant and Kut [12], they discover a region in the Mediterranean Sea that is a spatial outlier in 1998, where the years immediately preceeding and following it, 1997 and 1999, contain different values for the region. While this is also considered to be an spatio-temporal

outlier by our definition, we remove the limitations imposed by Birant and Kut's definition, and are also able to discover spatio-temporal outliers that persist over several time periods and which may move or evolve in shape and size. This is important as valuable outliers such as extreme events may exist over a number of time periods.

Before moving outliers can be discovered, however, we first need to find the spatial outliers in each time period. To identify the most significant outliers, we use a measure known as discrepancy. One well-known method for determining the discrepancy of spatial outliers is the spatial scan statistic, first introduced in [15]. This statistic has been applied by [10] to spatial grid data sets using an algorithm called Exact-Grid. We extend these to find the top-k outlier regions, rather than only finding the single 'highest discrepancy' region.

The Exact-Grid algorithm finds every possible different rectangular region in the data using four sweep lines to bound them. Once found, a well-known spatial scan statistic known as Kulldorff's scan statistic is applied to give each rectangular region a discrepancy value that indicates how different it is from the rest of the dataset [1].

The Kulldorff spatial scan statistic uses two values: a measurement m and a baseline b. The measurement is the number of incidences of an event, and the baseline is the total population at risk. For example, when finding disease outliers, m would be the number of cases of the disease and b would be the population at risk of catching the disease [10].

To calculate the Kulldorff scan statistic, $d(m, b, R)$ for a region R, we first need to find the measurement M and baseline B values for the *whole* dataset, where $M = \Sigma_{p \in U} m(p)$ and $B = \Sigma_{p \in U} b(p)$, and U is a box enclosing the entire dataset.

We then use these global values to find m and b for the local region R, by letting $m_R = \Sigma_{p \in R} \frac{m(p)}{M}$ and $b_R = \Sigma_{p \in R} \frac{b(p)}{B}$.

Once these values have been found, all we need to do is perform a simple substitution into the Kulldorff scan statistic, which is given by:

$$d(m_R, b_R) = m_R log(\frac{m_R}{b_R}) + (1 - m_R) log(\frac{1 - m_R}{1 - b_R})$$

if $m_R > b_R$ and 0 otherwise.

One of the most notable advantages of using the spatial scan statistic is that its ability to detect outliers is *unaffected by missing data regions*. This is particularly relevant in geographical data, which often contains a large number of missing values for regions and/or time periods.

For the Exact-Grid algorithm, one approach of dealing with missing values was to set missing value as the average value in the dataset. That would mean that the measurement m for the grid cell would be 0, while the baseline b for the grid cell would become 1. However, this large baseline value has an impact on the data, and causes larger grids to be selected as high discrepancy regions due to the larger baseline population considered to be at risk. Instead, the aproach we have adopted for each missing grid, is to set the baseline value b, which represents the total population, to zero, as there is no population present in a

	1	2	3	4
1	1	1	0	0
2	1	1	0	0
3	0	0	0	1
4	0	0	1	0

Fig. 3. An example grid for calculating Kulldorff's scan statistic

missing region. In addition because the population is zero, the measurement m must be zero. This means that when we try to calculate the spatial scan statistic, $m_R > b_R$ does not hold so the resulting value is 0. As shown in the following example, zero regions are not considered as outlier regions, and so the presence of missing values has minimal effect on the ability of the spatial scan statistic to find outliers. In essence we are able to ignore the missing regions while still detecting valuable and interesting outliers.

An example of the application of the Kulldorff scan statistic is provided in Figure 3. In this example, the maximum discrepancy of the shaded area is calculated by finding $M = 6$, $B = 16$, $m_R = \frac{4}{6}$ and $b_R = \frac{4}{16}$. Substituting this into the formula gives a value of $d(m_R, b_R) = 0.3836$.

Previous work using the spatial scan statistic to detect space-time clusters in point data has been done in [16]. They use a pyramid with square cross sections at each time interval, that can either expand or contract from the start to finish of the time interval. Candidate clusters are generated in a biased random fashion using a randomised search algorithm. However, their algorithm aims to discover clusters of point data whereas our research focuses on the detection of outliers from grid data. We have adopted the concept of using rectangular cross-sections at particular time periods from [16] to finding outliers. However, the methods we have used to generate the initial rectangles, and also the subsequent rectangles are different. These changes allow us to accommodate the discovery of outliers from data located in a spatial grid, and to find all sequences of rectangular outliers that are stationary, move and/or change shape. Due to the size and dimensionality of the data, finding patterns which deviate from the normal behaviour of the dataset is a non-trivial problem. While previous methods exist to find spatial outliers in spatio-temporal data, to the best of our knowledge, no other method exists that is capable of finding spatial outliers over more than one time period.

4 Our Approach

This section describes our approach taken to discover sequences of spatial outliers over time. It consists of three main steps:

1. Find the top-k outliers for each time period, using the **Exact-Grid Top-k** or **Approx-Grid Top-k** algorithms.

2. Using these top-k for each time period, find all the sequences of outliers over time and store into a tree, using the **Outstretch** algorithm.
3. Extract all possible sequences from the tree using the **RecurseNodes** algorithm.

The resulting output from the above steps is a list of all sequences and subsequences of outliers found in the dataset. Each of these steps is described in the following subsections.

4.1 Exact-Grid Top-k

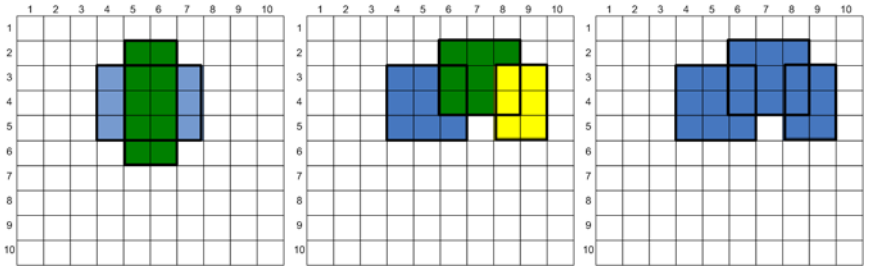
The Exact-Grid algorithm was proposed by Agarwal *et al.* [10]. It uses 4 sweep lines to find all possible different shaped regions that are located over a grid space of size $g \times g$. It takes $O(g^4)$ time to run the algorithm, since there are g^4 rectangles to consider. Runtime is minimised by maintaining a count of the measurement m and baseline b values for each row between the left and right scan lines. By doing this they are able to calculate the Kulldorff discrepancy value in constant time. Our extension to the Exact-Grid algorithm, called Exact-Grid Top-k, finds the top-k outliers for each time period. This is important since there may be a number of interesting outliers present in any given time period.

Since the Exact-Grid algorithm only finds the single highest discrepancy outlier, it did not have to take into account overlapping regions, as any region which had a lower discrepancy was simply replaced. When adding regions to the list of top-k algorithms however, we need to consider the case where there are overlapping regions, or else we could end up with a list of top-k regions that lie over the same area. This is illustrated in Figure 4(a), where the green region is overlapping the blue region. The different types of overlap that we considered are shown in Figure 5.

However, simply eliminating the lowest discrepancy region is not always the best solution, particularly if the regions only overlap slightly as this could eliminate some potentially interesting outliers. Therefore, we have introduced a threshold parameter, so we can specify the maximum amount of allowable overlap between regions.

Another issue is the scenario where there is a chain of overlaps, as shown in Figure 4(b). In this case, the discrepancy of the blue region is less than that of the green, and the discrepancy of the green is less than the yellow (i.e. $d(blue) < d(green) < d(yellow)$). If we are eliminating based on the highest discrepancy, and we find the blue region first and add it to our list of top-k outliers, when we find the green region, it will replace the blue region in the top-k outlier list. Then, when we find the yellow region, it will replace the green region in the list of top-k outliers. This creates a chain effect, which is problematic as the blue region may be quite different or far from the yellow region and yet has been eliminated.

One option that was considered was to form a union between the two regions. Then if the discrepancy of the unioned region was higher, the two sub-regions would be discarded, and the union would be stored in their place in the list of



(a) Two-region overlap problem (b) Chain overlap problem (c) The union solution to the overlap problem

Fig. 4. The overlap problem

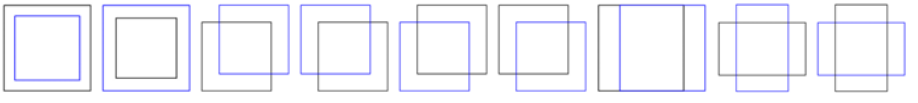


Fig. 5. Possible overlap types between two top-k regions

$top-k$. This concept is shown in figure 4(c). However, this would have decreased the efficiency of our algorithm, since it would be more complex to search irregular shapes for overlaps.

Instead, to deal with the overlap problem, we chose to allow some overlap between regions. This means we are able to include both spatial outliers in the top-k outlier list, which is important because despite their overlap, they represent significantly different locations. The amount of overlap is specified as a percentage, and the algorithm allows the user to vary this to the most appropriate amount for their particular application domain. The procedure is described in the following paragraphs.

The Exact-Grid Top-k algorithm finds the top-k outliers for each time period by keeping track of the highest discrepancy regions as they are found. As it iterates through all the region shapes, it may find a new region that has a discrepancy value higher than the lowest discrepancy value ($k^{th}value$) of the top-k regions so far. We then need to determine if this region should be added to the list of top-k regions. To do this we need to determine the amount of overlap that the new region has with regions already in the top-k.

For any top-k region that this new candidate region overlaps with, we first calculate the percentage overlap between the two regions. If it overlaps more than the percentage specified by the parameter at input, such as 10%, then we compare the discrepancy values of the two regions. If the new region has a higher discrepancy value, it will replace the other region, otherwise the new region will not be added. In the case where the percentage overlap is below the parameter specified maximum allowable overlap, the region will added to the list of top-k, provided that it does not violate the overlap condition with any of the other top-k regions. The Exact-Grid Top-k algorithm is shown in Algorithm 1.

Exact-Grid Top-k computes the overlapping region in $O(k)$ using the subroutine in Algorithm 2, since it has to check the new potential top-k region against all previous regions for overlap. Because of this, the total time required by the algorithm is $O(g^4k)$.

Algorithm 1. Exact-Grid Top-k Algorithm

Input: $g * g$ grid with values $m(i,j)$, $b(i.,j)$, $max\_overlap$
Output: Top-k highest discrepancy regions $topk$

```
1: // Bottom Sweep Line
2: for i = 1 to g do
3:    Initialize m[y] = m(i,y), b[y] = b(i,y) for all y
4:    for y = 2 to g do
5:       m[y]+ = m[y − 1], b[y]+ = b[y − 1]
6:    end for
7:    // Top Sweep Line
8:    for j = i+1 TO g do
9:       m = 0, b = 0
10:      for y = 1 to g do
11:         m+ = m(j,y), b+ = b(j,y),
12:         m[y]+ = m, b[y]+ = b
13:      end for
14:      // Left Sweep Line
15:      for k = 1 to g do
16:         // Right Sweep Line
17:         for l = k to g do
18:            if k = 1 then
19:               m = m[k],b = b[k]
20:            else
21:               m = m[l] − m[k − 1],
22:               b = b[l] − b[k − 1]
23:            end if
24:            if (d(m,b) > topk(k)) then
25:               c=the current region,
26:               topk=update_topk(c,topk)
27:            end if
28:         end for
29:      end for
30:   end for
31: end for
```

The Update Top-k subroutine calls the **get_overlap** method, which calculates the percentage overlap between region c, the current region under examination and each of the regions $topk$ in the list of top-k regions. If the overlap is less than 10%, the region will be added to $topk$ and will bump the k^{th} highest discrepancy region off the list. Otherwise only the highest discrepancy region will be kept in $topk$.

Algorithm 2. Update Top-k Subroutine

Input: c, $topk$, $max\_overlap$
Output: Top-k highest discrepancy regions $topk$

```
1: for all tk = topk do
2:    ov=get_overlap(c, tk)
3:    if ov < max_overlap then
4:       add c to topk
5:    else
6:       if dval(c)>dval(tk) then
7:          replace tk with c in topk
8:       end if
9:    end if
10: end for
```

4.2 Approx-Grid Top-k

As shown in Figure 3, the main difference between Exact-Grid Top-k and Approx-Grid Top-k is their approach to creating the test rectangles from the grid. The two algorithms both use two horizontal lines to determine the bottom and top edges of the test area. However, while Exact-Grid uses two vertical lines (lines 14-17 in Algorithm 1) to form the left and right edge, Approx-Grid Top-k projects all points inside the two horizontal lines onto the bottom horizontal sweep line (lines 6-8 in Algorithm 3).

By doing this, Approx-Grid Top-k reduces the maximising problem down to one-dimension. Thus, instead of two sweep lines moving from left to right, it simply finds the preceding interval r which maximises the above function (line 11 in Algorithm 3). Then the ends of this interval are used as the left and right edges of the test rectangle (line 12 in in Algorithm 3).

This is done using the same linear function as in Agarwal's paper [10], which is:

$$L(m_R, b_R) = \cos(\sin(\pi^2/8))m_R - \sin(\sin(\pi^2/8))b_R$$

where R is the bottom sweep line. From this we can find the interval r on the bottom sweep line that maximises the linear function.

To use the linear function, we let the shortest interval on a sweep line (a single grid cell) be a 'unit'. Then we:

- Go through each unit along the bottom sweep line to find the unit with the highest linear function value;
- Extend this unit left and right one unit at a time until the addition of one unit does not result in a higher linear function value;
- Then return these continuous units as the maximum linear discrepancy ínterval range.

The runtime of Approx-Grid Top-k is $O(g^3k)$. This includes g iterations for the left sweep line, g iterations for the right sweep line, g iterations for finding the maximized interval and k iterations for the **update_topk** routine.

4.3 The Outstretch Algorithm

To discover sequences of outliers, we developed Outstretch, which is detailed in Algorithm 4. Outstretch takes as input the top-k values for each year period under analysis, and a variable r, the region_stretch, which is the number of grids to 'stretch' by on each side of an outlier. This is shown in Figure 6.

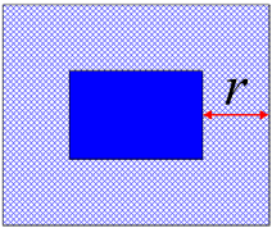


Fig. 6. Region Stretch Size r

Outstretch then examines the top-k values of the second to last available year periods. For all the years, each of the outliers from the current year are examined to see if they are framed by any of the stretched regions from the previous year, using the function **is_framed**. If they are, the variable 'framed' will return true, and the item will be added to the end of the previous years child list. As a result, all possible sequences over all years are stored into the *outlier_tree*.

An example tree structure is shown in Figure 7(a). In this example we have found the top 4 outliers over 3 time periods. These are labelled with two numbers, the first being the time period, the second being an identification number of the outlier from that time period. The algorithm stores all single top-k outliers from each time period, *yrly_topkvals*, as rows in a table, where each of these rows contains the outliers children. An example table corresponding to the example tree in Figure 7(a) is given in Figure 7(b). From this table we can extract the spatial outliers that persist over several time periods in a similar location.

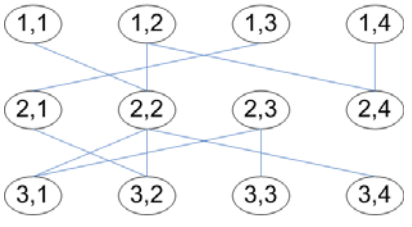
The example in Figure 8 shows how a sequence of outliers over three time periods can be collected. In each diagram, the outlier is represented by the solid blue region, while the stretch region is represented by the shaded blue region. In this example, the stretch size r equals 1. To begin, the outlier from the first time period is found. Then the region is extended by the stretch size on all sides, and is searched for outliers that are enclosed by it in the following year. If one is found that lies completely within the stretch region, a new stretch region is generated around the new outlier. This new stretch region is then searched for outliers in the third time period. This process continues until all time periods have been examined or there are no outliers that fall completely within the stretch region. As each of these outliers sequences are discovered they are stored into a tree.

The Outstretch algorithm runs in $O(n^3)$, since for each of the time periods available it iterates through all the top-k outliers for that period, and compares them against all the outliers from the previous time period.

Algorithm 3. Approx-Grid Top-k Algorithm

Input: $g \times g$ grid with values $m(i,j)$, $b(i,j)$, $max\_overlap$
Output: Top-k highest discrepancy regions $topk$

```
1: // Bottom Sweep Line
2: for i = 1 to g do
3:    Initialize m[y] = m(i,y), b[y] = b(i,y) for all y
4:    // Top Sweep Line
5:    for j = i+1 TO g do
6:       for y = 1 to g do
7:          m[y] = m[y] + m(j,y),
8:          b[y] = b[y] + b(i,y)
9:       end for
10:      // The interval that maximizes the linear function
11:      r = arg max_{r∈R} L(r),
12:      (y_b, y_t) = r
13:      if y_b = 1 then
14:         m = m[1],b = b[1]
15:      else
16:         m = m[y_t] − m[y_b − 1],
17:         b = b[y_t] − b[y_b − 1]
18:      end if
19:      if (d(m,b) > topk(k)) then
20:         c=the current region,
21:         topk=update_topk(c,topk)
22:      end if
23:   end for
24: end for
```



(a) An example outlier tree built by the Outstretch algorithm

Outlier	Number of Children	Child List
(1,1)	1	(2,2)
(1,2)	2	(2,2),(2,4)
(1,3)	1	(2,1)
(1,4)	1	(2,4)
(2,1)	1	(3,2)
(2,2)	3	(3,1),(3,2),(3,4)
(2,3)	2	(3,1),(3,3)
(2,4)	0	
(3,1)	0	
(3,2)	0	
(3,3)	0	
(3,4)	0	

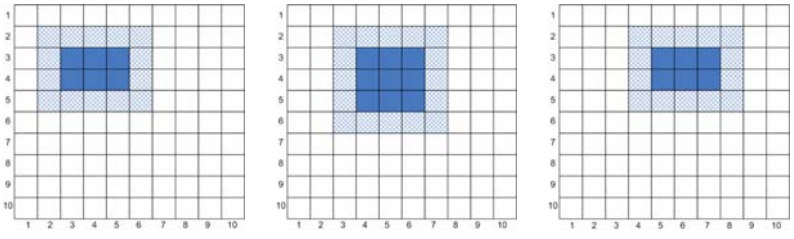
(b) The outlier table corresponding to the outlier tree in (a)

Fig. 7. An example outlier tree and table

Algorithm 4. Outstretch Algorithm

Input: *yrly_topkvals*, region stretch (r), years (y)
Output: outlier_tree (tr)

```
1: for yr = 2 to y do
2:    c = yrly_topkvals(yr)
3:    for all c do
4:       p = yrly_topkvals(yr-1)
5:       for all p do
6:          framed = is_framed(c,p,r)
7:          if framed = true then
8:             tr(p,len(tr(p))+1) = c
9:          end if
10:      end for
11:   end for
12: end for
```



(a) Outlier at t=1 (b) Outlier at t=2 (c) Outlier at t=3

Fig. 8. Example Outstretch Algorithm Sequence

4.4 The RecurseNodes Algorithm

From the tree structure, all possible sequences are extracted using a simple recursive algorithm, described in Algorithm 5. RecurseNodes takes 4 input variables. *outlier_tree* contains a list of each node and its children. *sequence* contains the sequence nodes so far, excluding the child value. *child_list* is a list of all the children of the last sequence item. *sequence_list*, is a list of all the sequences that have been generated from the outlier_tree at any point in time. The RecurseNodes algorithm has a running time of $O(n^y)$ where n is the total number of length-1 outliers, and y is the number of years of data and maximum possible length of an outlier sequence.

5 Experiments

5.1 Data

Special features of geographical data need to be considered when performing geographical data mining. Traditional data mining algorithms often assume the

Algorithm 5. RecurseNodes Algorithm

Inputs: outlier_tree(tr), sequence(seq), child_list(ch_list), sequence_list(seq_list)
Outputs: sequence_list (seq_list)

```
1: for all c in ch_list do
2:    new_seq = seq + c
3:    // append new_seq to the end of seq_list:
4:    seq_list(len + 1) = new_seq
5:    //get the grandchildren:
6:    gchildr = tr(c)
7:    if size(gchild) > 0 then
8:       seq_list =
9:       RecurseNodes(tr,new_seq,gchildr,seq_list)
10:   end if
11: end for
```

data to be independent and/or identically distributed [17]. These assumptions would violate Tobler's first law of Geography, which states that 'everything is related to everything else, but near things are more related than distant things' [18]. For the purpose of our experiments, we have removed some of the temporal dependencies in the data through deseasonalisation. This allows us to discover more interesting patterns.

Secondly, we have had to consider the effects of missing data. In our dataset, there are a large number of missing values, mostly in regions that do not lie over land masses. One of the advantages of using the spatial scan statistic is that it is able to discover significant outliers despite their close proximity to regions that contain missing data. Section 3 provides further details on the techniques used to handle missing values.

The data used in our experiments is the South American precipitation data set obtained from the NOAA [6]. It is provided in a geoscience format, known as NetCDF. A description of the data is provided in both [6] and [9]. The data are presented in a grid from latitude $60°S$ to $15°N$ and longitude $85°W$ to $35°W$. It contains daily precipitation values from around 7900 stations, whose values are averaged for each grid between 1940-2006.

Before we apply our algorithms to the data it must be pre-processed. It is difficult to spot interesting patterns in raw precipitation values as rainfall patterns are seasonal, so our attention is first drawn to these seasonal patterns. Such patterns are not very interesting, as we are usually aware of which seasons have greater rainfall. A more interesting statistic is the deviation of precipitation from the normal amount of precipitation expected at any particular time in the year. Data in this form is deseasonalised, and the deseasonalisation process is described in [9].

Once we have completed the deseasonalisation procedure, we take the average of the deseasonalised values over each period, for each grid, before running either the Exact-Grid Top-k or Approx-Grid Top-k algorithm, which gives a discrepancy value to each region.

5.2 Experimental Setup

The effectiveness of the Outstretch algorithm is evaluated by counting the total number of spatio-temporal outliers in the outlier tree, and the length of these outliers.

For this experiment, we set the input variables as shown in Figure 9(a).

Allowable Overlap is the maximum allowable size of two overlapping regions. This means that if two regions are overlapping by less than the percentage specified, we will not replace the one with the lower discrepancy value with the other.

Number of top-k is the maximum number of high discrepancy regions that are to be found by the Exact-Grid Top-k and Approx-Grid Top-k algorithms.

Extreme Rainfall sets the threshold percentage that the deseasonalised average rainfall must exceed to be considered extreme. In our experiments we chose the 90th percentile of values. This means that given deseasonalised average rainfall for all regions in South America, we consider only those regions whose rainfall was significantly different from the mean of all regions by more than 90%.

Region Stretch describes the number of grids to extend beyond the outlier to check for outliers that are bounded by it in the subsequent time period, as shown in Figure 6.

Figure 9(b) describes the subset of South American precipitation data that we use in our experiments from [6].

Variable	Value
Allowable Overlap	10%
Number of top-k	5
Extreme Rainfall	90%
Region Stretch	5

(a) Variable Setup

Variable	Value
Num Year Periods	10
Year Range	1995-2004
Grid Size	2.5°x2.5°
Num Latitudes	31
Num Longitudes	23
Total Grids	713
Num Data Values	2,609,580

(b) Subset of Precipitation Data Used

Fig. 9. Experimental Setup

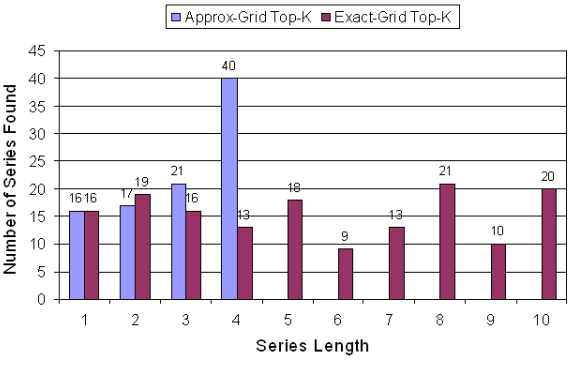
5.3 Experimental Results

The application of our algorithm to South American precipitation data involved three steps described in Section 4.

The first stage of our algorithm involved finding the top-k outliers at each time step using the Exact-Grid Top-k and Approx Grid Top-k algorithms, so we can compare the two techniques. The results of this are shown in Figure 10(a), which show that when we set $k = 5$, we were able to find 5 top outlier regions for all years using the Exact-Grid Top-k algorithm and usually less than 5 top outlier regions using the Approx-Grid Top-k algorithm.

Year	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004
Exact-Grid Top-k	5	5	5	5	5	5	5	5	5	5
Approx-Grid Top-k	5	5	5	5	5	3	2	3	2	3

(a) Outliers found each year



(b) Length and number of outliers found over all years

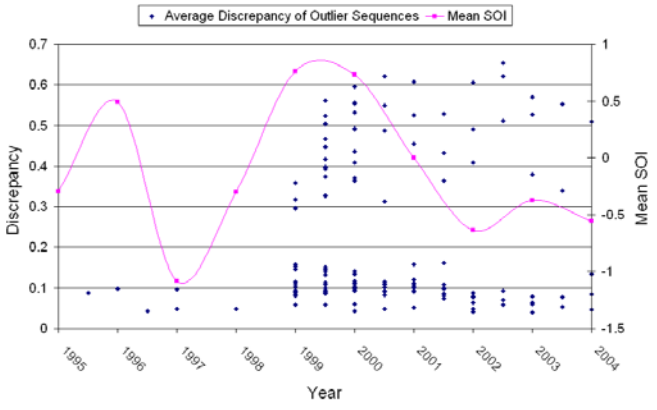
Fig. 10. Outliers found

From the second and third stages of our algorithm, we found a total of 155 outlier sequences over 10 years when using the Exact-Grid Top-k algorithm and 94 outlier sequences for the Approx-Grid Top-k algorithm. These sequences ranged from a minimum length of 1 to a maximum length of 10, since there are 10 year periods in our data subset. The results of this are summarised in Figure 10(b).

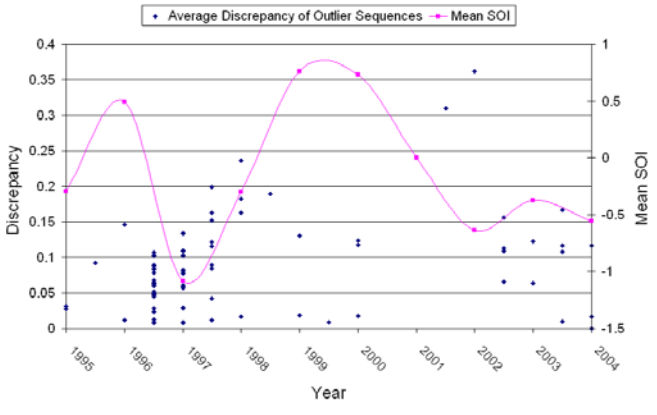
Figures 11(a) and (b) show the mean discrepancy for the outlier sequences generated by the Exact-Grid Top-k and the Approx-Grid Top-k algorithms respectively. The discrepancy is plotted at the middle year of the sequence. For example, given a sequence from 1999 to 2001, the mean discrepancy for the points in the sequence will be plotted on the graphs as a point for the year 2000. Both graphs also show the mean SOI for each year, so we are able to visualise the relationship between the discrepancy of the outlier sequences and the mean SOI for the same period.

As noted previously, a negative SOI indicates that there was an El Niño event at that time. In addition, the more negative the SOI, the stronger the El Niño event [19].

In Figure 11(a), we can see that between 1999 and 2004 there are a large number of outliers with a higher than average mean discrepancy value. Many of these outliers are plotted at the midpoint of long 9 or 10 year sequences whose start is 1995 and end is 2004, and so appear plotted around 1999-2000. These outliers are not as significant because they occur over all 10 periods and so are consistently outliers in each year. Therefore, if we only consider the outliers which are not plotted at the 1999-2000 time periods, we can see that there are



(a) Mean discrepancy of Exact-Grid Top-k sequences and the mean SOI



(b) Mean discrepancy of Approx-Grid Top-k sequences and the mean SOI

Fig. 11. Relationship between discrepancy and SOI

a large number of high discrepancy outliers plotted around 2002. This coincides with an El Niño event. Although this event is much weaker than that in 1997, there are many more higher discrepancy outlier sequences whose year midpoint lies during this time. From this one conclusion we could draw is that during weaker El Niño periods, there are more extreme-extreme outlier events.

This trend can also be seen in Figure 11(b), where there are two extremely high discrepancy outliers during the 2002 El Niño event, and more moderate outlier occurrences during the 1997 El Niño event.

To evaluate the computational performance of Approx-Grid Top-k and Exact-Grid Top-k, we ran both algorithms over 1995 to 2004. This is shown in Figure 12, where we can see that the Approx-Grid Top-k algorithm is much faster.

Exact-Grid Top-k	Approx-Grid Top-k
229s	35s

Fig. 12. Time taken to discover the Top-k regions over 10 years

6 Discussion and Conclusion

Exact-Grid Top-k and Approx-Grid Top-k are able to find the top-k high discrepancy regions from a spatial grid. We have shown that Approx-Grid Top-k finds a similar number of outliers from each time period, significantly faster than the Exact-Grid Top-k algorithm by approximating the discrepancy of the outlier regions. While some longer sequences could not be found by Approx-Grid Top-k, shorter sequences may be able to convey the same trend that longer sequences do. This can be seen in Figures 11(a) and (b) where the general trend shown against the SOI is similar for both algorithms.

We have also extended this algorithm to include the ability to discover moving spatio-temporal outlier sequences that change location, shape and size, using our Outstretch algorithm, which stores the found sequences into a tree structure. To then extract sequences from the tree, we have provided the RecurseNodes algorithm.

Our results demonstrate the successful application of our algorithm to a large precipitation data set. Given the large size of our dataset, the time taken to run the algorithm is quite good. We have shown that our algorithm is capable of finding spatial outlier sequences and subsequences that occur over several time periods in the South American precipitation data. In addition, we have shown one possible way of interpreting the results by comparing the behaviour of the outlier regions to the El Niño and La Niña phases of the ENSO phenomenon.

Future work could adapt our approach to find spatio-temporal outliers in point data. In [10], the authors provided an algorithm called Exact to discover high discrepancy regions in point data. Additionally, with enough historical data, future work could use the historical patterns we have discovered to predict the timing, location and duration of future events.

References

1. Agarwal, D., Phillips, J.M., Venkatasubramanian, S.: The Hunting of the Bump: On Maximizing Statistical Discrepancy. In: Proceedings 17th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1137–1146. ACM Press, New York (2006)
2. Han, J., Altman, R.B., Kumar, V., Mannila, H., Pregibon, D.: Emerging Scientific Applications in Data Mining. Communications of the ACM 45(8), 54–58 (2002)
3. Miller, H.J.: Geographic Data Mining and Knowledge Discovery. In: Wilson, Fotheringham, A.S. (eds.) Handbook of Geographic Information Science (2007)
4. Miller, H.J., Han, J.: Geographic Data Mining and Knowledge Discovery: An Overview In Geographic Data Mining and Knowledge Discovery. Taylor & Francis, New York (2001)

5. Openshaw, S.: Geographical Data Mining: Key Design Issues. In: Proceedings of GeoComputation (1999)
6. Liebmann, B., Allured, D.: Daily precipitation grids for South America. Bulletin of the American Meteorological Society 86, 1567–1570 (2005)
7. DAleo, J.S., Grube, P.G.: The Oryx Resource Guide to El Niño and La Niña. Oryx Press, CT (2002)
8. National Oceanic and Atmospheric Administration (NOAA) Climate Prediction Center: Monthly Atmospheric & SST Indices, `http://www.cpc.noaa.gov/data/indices` (Accessed February 2, 2008)
9. Wu, E., Chawla, S.: Spatio-Temporal Analysis of the relationship between South American Precipitation Extremes and the El Niño Southern Oscillation. In: Proceedings of the 2007 International Workshop on Spatial and Spatio-temporal Data Mining. IEEE Computer Society, Washington (2007)
10. Agarwal, D., McGregor, A., Phillips, J.M., Venkatasubramanian, S., Zhu, Z.: Spatial Scan Statistics: Approximations and Performance Study. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 24–33. ACM Press, New York (2006)
11. Cheng, T., Li, Z.: A Multiscale Approach for Spatio-Temporal Outlier Detection. Transactions in GIS 10(2), 253–263 (2006)
12. Birant, D., Kut, A.: Spatio-temporal outlier detection in large databases. In: 28th International Conference on Information Technology Interfaces, pp. 179–184 (2006)
13. Ng, R.: Detecting Outliers from Large Datasets In Geographic Data Mining and Knowledge Discovery, pp. 218–235. Taylor & Francis, New York (2001)
14. Theodoridis, Y., Silva, J.R.O., Nascimento, M.A.: On the Generation of Spatiotemporal Datasets. In: Proceedings of the 6th International Symposium on Advances in Spatial Databases, pp. 147–164. Springer, London (1999)
15. Kulldorff, M.: A Spatial Scan Statistic. Communications in Statistics - Theory and Methods 26, 1481–1496 (1997)
16. Iyengar, V.S.: On Detecting Space-Time Clusters. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge Discovery and Data mining, pp. 587–592. ACM, New York (2004)
17. Chawla, S., Shekhar, S., Wu, W., Ozesmi, U.: Modelling Spatial Dependencies for Mining Geospatial Data: An Introduction. In: Geographic Data Mining and Knowledge Discovery, pp. 131–159. Taylor & Francis, New York (2001)
18. Tobler, W.R.: A computer model simulation of urban growth in the Detroit region. Economic Geography 46(2), 234–240 (1970)
19. Redmond, K.: Classification of El Niño and La Niña Winters, `http://www.wrcc.dri.edu/enso/ensodef.html` (Accessed October 24, 2008)