CURIO : A Fast Outlier and Outlier Cluster Detection Algorithm for Large Datasets[*]

Aaron Ceglar, John F. Roddick and David M.W. Powers

School of Informatics and Engineering,
Flinders University,
PO Box 2100, Adelaide,
South Australia 5001.

Abstract

Outlier (or anomaly) detection is an important problem for many domains, including fraud detection, risk analysis, network intrusion and medical diagnosis, and the discovery of significant outliers is becoming an integral aspect of data mining. This paper presents CURIO, a novel algorithm that uses quantisation and implied distance metrics to provide a fast algorithm that is linear for the number of objects and only requires two sequential scans of disk resident datasets. CURIO includes a novel direct quantisation technique and the explicit discovery of outlier clusters. Moreover, a major attribute of CURIO is its simplicity and economy with respect to algorithm, memory footprint and data structures.

Keywords: Outlier, Anomaly Detection, Outlier Clustering

1 Introduction

Outlier (or anomaly) detection is an important problem for many domains. Within medicine, for example, it is commonly the exceptions that provide the insight into a problem. Although rare events are by definition infrequent, their significance is high compared to other events, making their detection important. Outlier detection is a mature field of research with its origins in statistics (Markou & Singh 2003*a*). Current techniques typically incorporate an explicit distance metric, which determines the degree to which an object is classified as an outlier. A more contemporary approach incorporates an implied distance metric, which obviates the need for the pairwise comparison of objects. This enables the efficient analysis of very large (disk resident) datasets as memory residency of all objects for pairwise comparison is not required. This implied approach typically uses domain space quantisation (Knorr & Ng 1998, Papadimitriou et al. 2003, Chiu & Fu 2003, Chaudhary et al. 2002) enabling distance comparisons to be made at a higher level of abstraction and, as a result, obviates the need to recall raw data for comparison purposes. CURIO adopts this implied or abstract approach, proposing a novel quantisation and object allocation technique that enables both to be undertaken in a single direct step.

Through these principles and the use of dynamic data structures, CURIO provides a novel algorithm that discovers outliers in large disk resident datasets in two sequential scans, which is comparable with current best practice.

Importantly, there is an industry need to discover not only outliers but also outlier clusters. For example, in medicine, the investigation of what appear to be anomalous groupings have frequently yielded insight to a problem (Roddick et al. 2003). By clustering similar (close) outliers and presenting cluster characteristics it becomes easier for users to understand the common traits of similar outliers, assisting the identification of outlier causality. Although proposed as an area of further work by Knorr (2002), the realisation of this functionality is novel and enhances the utility of the CURIO algorithm.

This paper is organized as follows. Section 2 discusses previous outlier discovery research. Section 3 introduces the theory behind CURIO and Section 4 presents the CURIO algorithm. Section 5 presents a comparative discussion including experimental results and also introduces the prototype. Finally, Section 6 presents a short conclusion and areas of further work.

2 Previous Work

Outlier detection algorithms are founded upon statistical modeling techniques, either predictive or direct. Predictive techniques use labelled training sets to create an outlier data model (within which outliers fall) for a domain, which is subsequently used to classify new data objects. Direct techniques, which include statistical, clustering, deviation, proximity and density based techniques, refer to those in which labelled training sets are unavailable and therefore the classification of objects as outliers is implemented through the specification of statistical heuristics. Although typically more complex than predictive techniques, direct methods are less constrained as discovery is not dependent upon pre-defined models. This section provides a summary of previous outlier detection research with a focus upon direct methods. A recent, comprehensive survey of outlier detection methodologies is presented by Hodge & Austin (2004).

Supervised neural networks and decision trees are two common forms of predictive outlier analysis. Supervised neural networks use the target class of the labelled training set to drive the learning process, adjusting weights and thresholds to ensure that the network can correctly predict the class of new unclassified objects. While neural networks adjust well to unseen patterns and are capable of learning complex class boundaries, the training set must be traversed many times to allow the network to settle, in order to provide an accurate model. Hawkins et al. (2002) and Markou & Singh (2003*b*) provide surveys regarding neural networks in outlier detection. While super-

[*]We are indebted to the staff at PowerSolutions Pty Ltd with whom this algorithm was developed and tested as part of the PKB system funded, in part, by an AusIndustry START grant.

vised neural networks require numerical data, Skalak & Rissland (1990) propose the use of decision trees as a robust and scalable method to identify simple class boundaries in categorical data.

Early statistical approaches construct probability distribution models under which outliers are objects of low probability. These *discordancy tests* (Barnett & Lewis 1994) are typically univariate and require ordinal data, however some multivariate extensions have been proposed (Mahalanobis et al. 1949). More complex statistical outlier tests have been proposed, including the use of adaptive nominators (Billor et al. 2000), information measures (Lee & Xiang: 2001) and convex peeling (Rousseeuw & Leroy 1996). While statistical methods have a solid foundation and are useful given sufficient knowledge of the data and the type of test to be applied, this is often not the case and therefore their practical use is limited. Furthermore, although multivariate techniques have been proposed, they are few and scale poorly. A good review of statistical techniques is provided by Markou & Singh (2003*a*).

Clustering aims to partition a dataset into groups that maximise intra-group similarity and inter-group dissimilarity. Clustering is also based upon distance metrics and while some existing algorithms such as CLARANS (Ester et al. 1995), DBSCAN (Ng & Han 1994) and BIRCH (Zhang et al. 1996) consider outliers (to the extent of ensuring that they do not interfere with the clustering process) others have been adapted to identify them, through post-processing (Jain et al. 1999) or algorithmic specialisation. For example, Nairac et al. (1999) extend K-means, Bolton & Hand (2001) extend K-medoids, while others (Yu et al. 2002, Struzik & Siebes 2002) use wavelet transforms to find outliers by progressively removing clusters from the dataset.

Deviation based algorithms inspect object characteristics, identifying outliers as those that deviate most from the identified standard features of a dataset. Arning et al. (1996) identify those objects that disturb a given time series as outliers, by labelling the subset of data that leads to the greatest reduction in entropy as outliers. Although this algorithm claims linear complexity, it depends upon the incorporation of an incrementally calculated dissimilarity function, which is often not possible.

Proximity or distance-based techniques identify outliers as those objects distant from most other objects. The naive complexity of this is quadratic due to pairwise comparison, however Bay & Schwabacher (2003) propose a simple pruning optimisation that reduces complexity given a randomly ordered dataset. Knorr & Ng (1998) propose DB-outliers which quantises the space into hypercells, achieving linear complexity in D ($D = |tuples|$) but exponential in κ ($\kappa = |dimensions|$), making the algorithm efficient for low dimensional spaces. Ramaswamy et al. (2000) propose an extension to DB-outliers that produces a ranked list of potential outliers and uses the micro clustering phase of BIRCH (Zhang et al. 1996) to quantise the space in near linear time. Both these algorithms use quantisation to narrow the search space and then use explicit distance metrics to identify outliers. A similar approach by Shekhar et al. (2001) examines point neighborhoods from a topologically connected, rather than distance based perspective.

To alleviate the curse of dimensionality, research has also been undertaken into the use of low-dimensional projections to identify outliers. Aggarwal & Yu (2001) adopt an evolutionary algorithm to discover those low dimension projections that are locally sparse and indicate the presence of outliers, while Angiulli & Pizzuti (2002) use Hilbert space filling curves to project the dataset multiple times into a linear representation, where each projection improves the accuracy of an object's outlier score in full-dimensional space.

While these extensions find global outliers, density based techniques, which are derived from density-based clustering algorithms, enable the discovery of local outliers, which are anomalies with respect to their local neighborhood. The seminal algorithm LOF (Breunig et al. 2000) calculates the Local Outlier Factor of each object through NN search, resulting in a complexity of at best $O(D\log(D))$. Jin et al. (2001) propose an optimisation by constraining LOF calculation to the top-n outliers using the concept of micro clusters from BIRCH (Zhang et al. 1996), which avoids the calculation of LOF for every object (given that $n \ll D$). Many extensions to LOF have been proposed, however of particular importance to this paper are those involving quantisation, namely aLOCI (Papadimitriou et al. 2003) and GridLOF (Chiu & Fu 2003) (see Section 5). Chaudhary et al. (2002) propose FastOut, a novel density based algorithm that quantises the domain using an extended form of κ-D trees (Bentley 1975) and ranks outliers based upon the inverse density ratio of its cell (the ratio between the size of the cell and the number of objects within it).

More recently, Tao et al. (2006) present the outlier detection method SNIF which is able to accommodate arbitrarily large datasets in three scans or the dataset through *prioritised flushing*. Priorities are assigned based on the likelihood that the object will be an outlier or a non-outlier with relatively few neighbours. Other work includes the RBRP (*Recursive Binning and Re-Projection*) binning technique (Ghoting et al. 2006) which iteratively partitions the dataset and thus limits comparisons of distance.

Building upon this previous work, CURIO presents a novel outlier detection algorithm that makes two significant contributions to the field. First, CURIO optimses the analysis of disk resident datasets by avoiding object-level comparisons and enabling analysis in only two sequential scans. Second, while previous techniques discover outliers and can imply outlier clusters through effective presentation techniques, CURIO explicity discovers these clusters, a novel functionality that increases usefulness. The following sections present the theory and implementation of CURIO, followed by a discussion that compares CURIO against its competitors, namely DB-outliers, FastOut, GridLOF and aLOCI.

3 Theory

Given the simple premise that outliers are distant from the majority of objects when represented in Euclidean space, if this κ-D space is quantised, outliers are those objects in relatively sparse cells, where the degree of relative sparsity is dictated by the parameter tolerance T. Given $T = 4$, Figure 1 presents a 2-dimensional grid illustrating both potential (grey) and valid (white labelled) outlier objects. However this simple approach can validate false positives as indicated by A, which is on the edge of a dense region. This problem can be resolved by either creating multiple offset grids or by undertaking a NN (Nearest Neighbour) search. Multiple offset grids requires the instantiation of many grids which are slightly offset from each other. This effectively alters the cell allocation of objects, and requires a subsequent voting system to determine if an object is to be regarded as an outlier. The alternative NN search explores the bounding cells of each identified potential outlier cell and if the number of objects within this neighbourhood exceeds T, all objects residing within the cell are eliminated from consideration. Both techniques were implemented, however the neighborhood search was found to be more accurate, and hence is the one presented.

This overarching theory provides the foundation of CURIO, enabling disk resident datasets to be analysed in two sequential scans. The quantisation and subsequent count based validation effectively discovers outliers (qv. Section 4), indicating that explicit distance threshold is not required and in fact often needlessly complicates the discovery process. CURIO incorporates an implied distance metric through cellular granularity, where the finer the granularity the shorter the implied distance threshold. This parameter, denoted precision, P, effectively quantises each dimension into 2^P equal length intervals, see Section 4. For example, Figure 2 illustrates the effect of increasing P by 1 (effectively doubling the number of partitions), resulting in potentially more outliers (i.e. G).

The provision of tolerance T and NN search enables the explicit discovery of outlier clusters. While the previous LOF algorithm (Breunig et al. 2000), purports to discover outlier clusters, they refer to an implied cluster identification, whereby a small group of objects significantly removed from the rest of the dataset are identified as **individual outliers** dependent upon the parameter *MinPts*. However they remain independent outliers, while CURIO provides a technique for explicitly grouping these independent outliers and presenting them as an explicit cluster.

Such functionality facilitates result interpretation by explicitly representing and differentiating the outlier clusters. Earlier techniques require the visual identification of outlier clusters by the user, and while this is satisfactory when $\kappa \leq 3$, as direct Euclidean mappings can be given, direct mappings cannot be applied at higher κ, making visual interpretation difficult. For example, given a 2D scatterplot presentation, what appears to be a single cluster, may be a diverse set of objects that happen to share commonality in the presented dimensions. Furthermore, the explicit identification of outlier clusters enables the derivation and subsequent presentation of cluster characteristics or traits to help identify causality.

CURIO identifies outlier clusters by constructing associative lists during outlier discovery, which effectively represents groups of similar valid outlier cells. These lists are then merged to resolve duplicates, ensuring that a valid outlier cell only appears in a single cluster. Due to the small number of outliers sought and their typical local sparsity, experimentation has shown this to be an effective way of identifying outlier clusters (qv. Section 4.3). An example of an outlier cluster is presented by the shaded region (indicating the valid neighbourhood) in Figure 2, consisting of the outlier points D, E, F and G.

4 CURIO

The CURIO algorithm consists of two phases, quantisation (the identification of candidate outlier cells) and discovery (the validation of these candidate cells). The following subsections present the algorithm and present the outlier cluster extension. While a normalised dataset is not required, it helps to eliminate skew and leads to better results.

4.1 Quantisation

CURIO's quantisation phase uses a novel technique to partition the domain and allocate objects as a single step. The storage structure S is realised as a hashtable for fast access and dynamic construction,

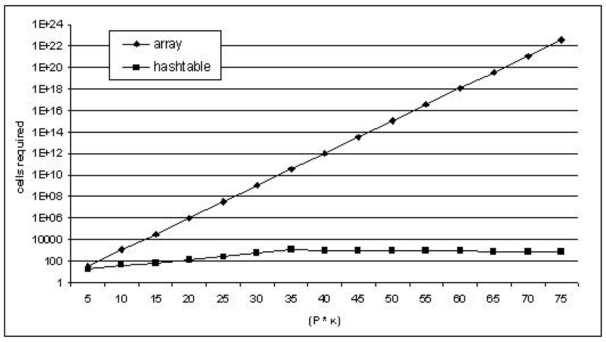


Figure 3: storage structure comparison

effectively representing a directly accessible abstraction of the hypercube, with each hash-bucket representing a cell.

The allocation of objects to cells is achieved by constructing the hash-index from the most significant P bits of each dimension, which are concatenated to form the cell or index identifier. For example, given that a three bit binary number (x) represents the range 0-7, the most significant bit represents whether $x \geq 4$, effectively quantising the single dimension space in two. Thus by taking the P most significant bits, 2^P partitions are constructed. For example P=5 provides 32 partitions across each dimension, and given 4 dimensions P=5 will result in the virtual creation of $2^{5*4} \approx 1$ million CURIO cells. Table 1 provides examples of index construction given 8 bit dimensions and P=3, with a space for clarity.

Given this indexing method, the discovery of candidate outlier cells requires a sequential scan of the dataset. The index is constructed, providing a hashtable key that identifies a relevant cell (hash bucket). Each cell then contains a count, which is incremented for each object mapped to it. Thus upon completion S contains the set of populated cells and their counts, with the set of candidate cells being those with $s^{count} \leq T : s \in S$.

A hashtable structure was found best suited to this situation given its dynamic nature. As precision or dimensionality increase, the number of potential cells grows exponentially ($2^{P*\kappa}$), resulting in increased domain sparsity, as the same objects are spread over more cells. Since only knowledge of the populated cells is required, static memory allocation for all domain cells, as required by an array structure A, is inefficient. Early experimentation considered the use of A for small κ, to provide direct access rather than the mapping required with hashtables, however the size of A and the time taken for its initial memory allocation was found to negate the benefit of direct access. Figure 3 shows the difference in required cell numbers for the UCI-KDD dataset (Hettich & Bay 1999) on internet usage data, while increasing P and κ. However it should be noted that an array structure is still reasonable given a dense dataset and coarse partitioning (number of cells $< 2^{24}$).

4.2 Outlier Discovery

The Outlier Discovery phase subsequently validates the potential outlier cells in S to find outlier objects. Each cell is validated by undertaking a NN search upon its first object access to eliminate any false positives. Given the inclusion of an abstract distance metric through quantisation, all objects within a valid outlier cell are outliers. Therefore an object is validated based upon its cell's classification.

By including neighbouring cells as part of the target cell's domain through NN search, false posi-

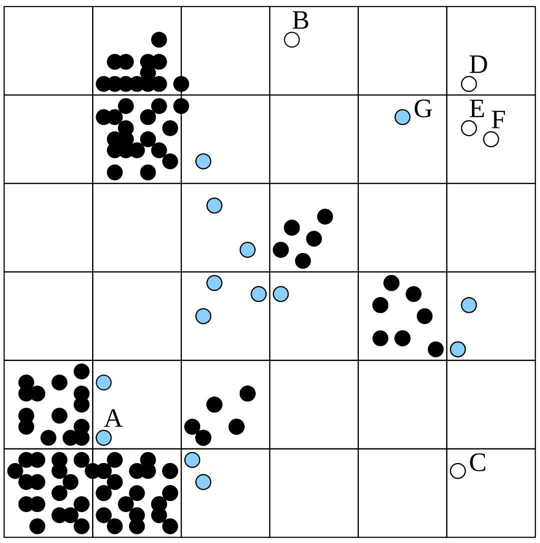


Figure 1: Example Grid

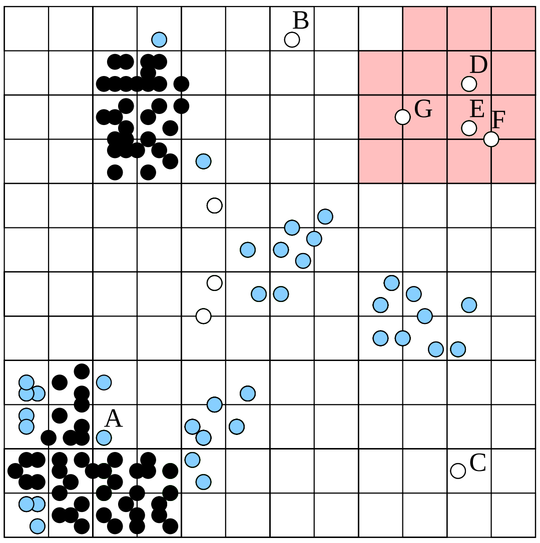


Figure 2: With increased precision

Dim 1	Dim 2	Dim 3	Index
001 00110	*101* 11100	*110* 00001	001 101 110
001 10110	*110* 01010	*011* 11001	001 110 011
100 00010	*000* 11001	*111* 01110	100 000 111

Table 1: Cell Identifier Construction

Algorithm 1 Discover Outliers

```
for all d ∈ D do
    I = calcIndex(d)
    cell s = S.get(I)
    if ! $s_{invalid}$ then
        if $s_{unknown}$ then
            $s_{valid}$ = exploreHood(S,$s_{count}$,I,N,T)
        end if
        if $s_{valid}$ then V.add(d)
    end if
end for
hoodIterator.initialise(I, N)
while hoodIterator.hasNext() do
    ct += S.get(hoodIterator.getNext())$_{count}$
    if ct > T then return false
end while
return true
```

tives are eliminated by comparing tolerance T against the domain count $s^{count}_{domain} = \sum\{\forall s^{count} : s \in neighbourhood\}$, rather than the target cell's count. The neighbourhood extent is defined by N, such that N defines the cellular radius of the NN search, hence $N = 1$ includes only bounding cells, while N=3 includes those up to 3 distant from the target cell. The search is inherently expensive, having a complexity of $O(2N+1)^{\kappa}$, typically N=1 has been found to perform well and reduces complexity to $O(3^{\kappa})$. However the search is optimised by terminating when $s^{count}_{domain} > T$.

Once a cell is validated (or classified) as either common or outlier, all objects pertaining to that cell are assigned the cell's classification, therefore the set of valid outliers V are those objects belonging to valid outlier cells. The outlier discovery process is presented in Algorithm 1.

4.3 Cluster Discovery

Cluster discovery requires an extension of the outlier discovery stage to construct associated outlier cell lists or cell clusters during cell validation. Figure 4 presents the outlier set from Figure 2 with cell

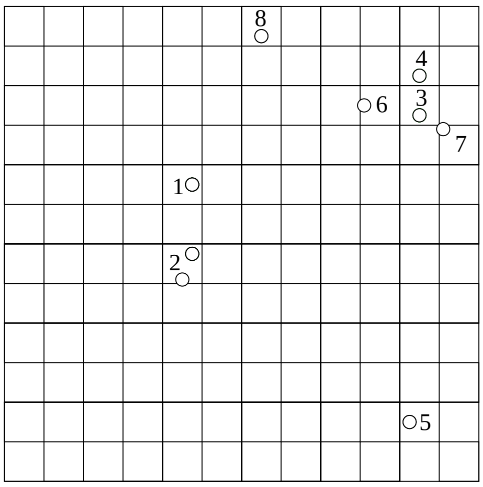


Figure 4: Valid Outlier cells

identifiers reflecting the order of cell validation. It is apparent that cells 1, 2, 5 and 8 have no associated valid cells and therefore form single cell clusters, while cells 3, 4, 6 and 7 form a multi-cell cluster. During validation, or NN search, each target cell is assigned a unique cluster identifier and upon encountering an earlier validated cell with a different cluster identifier, the later cluster is merged into the earlier cluster. Therefore upon completion if the target cell is validated, it also contains the required clustering information. For example, during validation, cell 1, 2 and 3 are validated, resulting the identification of $cluster_1 \Rightarrow 1$, $cluster_2 \Rightarrow 2$, $cluster_3 \Rightarrow 3$. Then cell 4 is validated, during which cell 3 is encountered, resulting in $cluster_3 \Rightarrow$ 3,4. Subsequently cell 6 and 7 are both validated, encountering members of $cluster_3$ and hence resulting in $cluster_3 \Rightarrow$ 3,4,6,7. This results in the eventual identification of 5 clusters : $\{1\}$, $\{2\}$, $\{3, 4, 6, 7\}$, $\{5\}$, $\{8\}$. The extended Outlier Discovery algorithm is shown in Algorithm 2 illustrating the generation of a cluster listing (#12) and cluster reduction through merging (#18).

Algorithm 2 Outlier Discovery with explicit Cluster Identification

```
 1: for all d ∈ D do
 2:    I = calcIndex(d)
 3:    cell s = S.get(I)
 4:    if ! $s_{invalid}$ then
 5:       if $s_{unknown}$ then
 6:          $s_{valid}$ = exploreHood(S,s,I,N,T)
 7:       end if
 8:       if $s_{valid}$ then s.add(d)
 9:    end if
10: end for
11: cluster l = new cluster(s)
12: clusterList.add(l)
13: hoodIterator.initialise(I, N)
14: while hoodIterator.hasNext() do
15:    cell c = S.get(hoodIterator.getNext())
16:    $s_{count}$ += $c_{count}$
17:    if $s_{count}$ > T then return false
18:    if $c_{valid}$ then merge($s_{cluster}$,$c_{cluster}$)
19: end while
20: return true
```

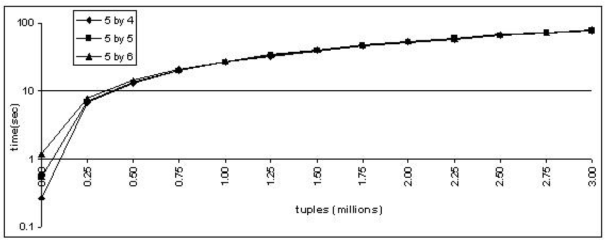


Figure 5: Dataset scalability (legend: Dimensions by Precision)

5 Discussion and Experiment Results

This section provides algorithmic comparison, experimental results and briefly introduces the prototype. CURIO is compared against the other fast outlier detection algorithms that incorporate quantisation, namely FastOut, aLOCI, GridLOF and DB-outliers (qv. Section 2), with respect to complexity, underlying data structures and disk resident analysis. The experimentation was conducted on a Pentium 2.4GHz with 1Gb of memory dedicated to the process and the code was written in Java 1.4.

CURIO has linear complexity with respect to D as illustrated in Figure 5, which is based upon the analysis of a real hospital administration dataset containing 3.1 million tuples and 5 selected analysis dimensions. Two sequential passes of D are required for analysis, resulting in $O(2D)$. Neighbourhood exploration of candidate cells is exponential with respect to κ, having a worst case of $O((2N+1)^{\kappa})$, however optimisation is provided through dynamic termination and the typical case of $N = 1$. Given this and the typically small number of candidate cells, the effect of this complexity upon performance becomes significant above ≈ 8 dimensions as shown in Figure 6, which is derived from the UCI-KDD Internet Usage Dataset (Hettich & Bay 1999). Experimentation has shown that for all practical purposes CURIO is efficient when $\kappa < 10$.

In comparison, GridLOF provides a pre-processing optimisation for LOF by quantising the domain and removing dense cell objects from further consideration. This requires NN search to identify *boundary* cells and is exponential with respect to κ. However GridLOF is also non-linear with respect to D given its subsequent LOF processing, which at best is $O(DlogD)$. Similar to CURIO, DB-outliers is exponential in κ (5^{κ}) and, although the authors claim DB-outliers is linear in D, object level pairwise comparison is required for a subset of objects and hence complexity with respect to D must be greater than linear.

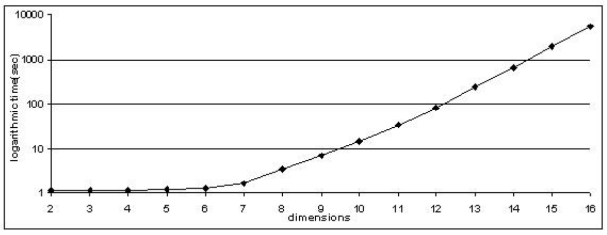


Figure 6: Dimensionality (κ) scalability

FastOut quantises the domain into regions of equal density and if the density ratio exceeds a specified threshold then the participant objects are outliers. FastOut claims linear complexity with respect to both κ and D. However valid outliers can occur near region boundaries and by not conducting a search of adjacent regions, false positives can occur. Therefore, although linear, FastOut is less accurate. aLOCI is theoretically similar to CURIO, being linear in D and requiring a NN search, however in practice it is almost linear with respect to κ, as it takes advantage of complex underlying data structures. aLOCI implements a forest of κ-D trees to quantise the domain, which recursively create regions of finer granularity. During discovery aLOCI identifies the relevant cell and undertakes NN search by rolling up the tree to a cell of suitable volume in which the target cell is centered. This coarse cell provides the NN boundary and the neighbourhood count is obtained by summing its subtree's cell's object counts. As this technique can introduce significant skew if no well fitting coarse cell can be found, aLOCI creates many, slightly offset, κ-D trees (10 to 30), and selects the best tree for each object's validation, subsequently improving result quality.

FastOut and aLOCI use variants of κ-D trees as their underlying data structures in comparison to the simple hashtables used by GridLOF, DB-outliers and CURIO. While κ-D trees provide efficient techniques for subtree accumulation, Knorr (2002) provides experimental evidence that the use of tree based indexing structures is typically uncompetitive due to hidden costs associated with the construction of these structures. This is exacerbated in aLOCI by the creation of a forest of trees.

Similar to CURIO, GridLOF and DB-outliers use hashtable storage structures. However the allocation process is not discussed in DB-outliers and hence a naive approach is assumed. While GridLOF mentions the construction of an index based upon the concatenation of interval id's for each dimension, it indicates a two stage process of interval calculation and allocation, and an intermediary mapping for each object. In contrast CURIO's quantisation technique accomplishes both interval calculation and allocation in a single direct step.

With regard to the analysis of disk resident datasets, CURIO and aLOCI are the most efficient as neither requires pairwise comparison nor the storing of objects within their underlying structures. Both GridLOF and DB-outliers require pairwise comparison and while GridLOF does not address the analysis of disk resident datasets, DB-outliers proposes a novel yet complex paginating algorithm to minimise disk access. Similarly, FastOut requires that all objects be stored within the κ-D tree to recursively divide cells evenly, ensuring that each new region is uniformly sparse. In contrast, both CURIO and aLOCI only require two sequential reads of a disk-resident dataset.

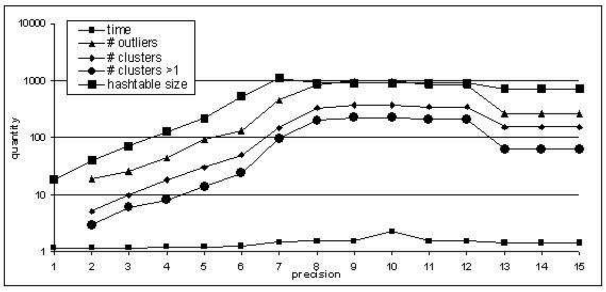


Figure 7: Effects of precision adjustment

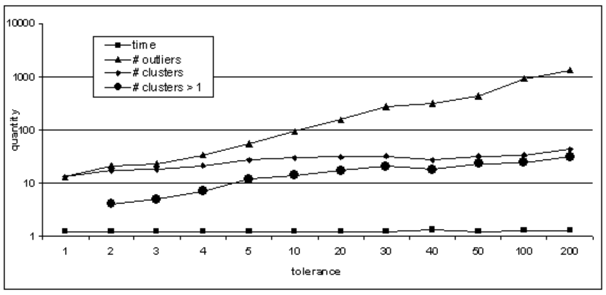


Figure 8: Effects of tolerance adjustment

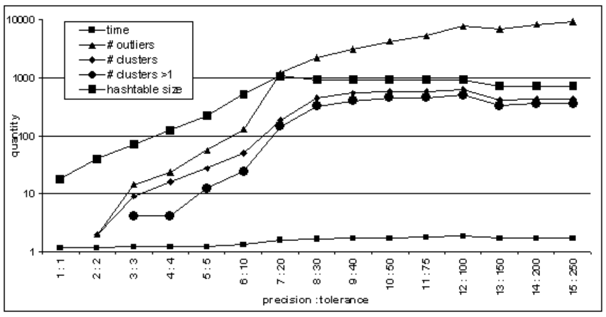


Figure 9: Effects of precision : tolerance adjustment

The CURIO algorithm has three parameters, precision P, tolerance T and neighbourhood extent N. While N is typically set to 1, the variance of both P and T affects the nature of results. This variance is illustrated in Figures 7, 8 and 9 which are derived from analysis upon the UCI KDD dataset (Hettich & Bay 1999) with a basis of $\kappa = 5$, P=5, T=10 and N=1. The set of scatterplot visualisations presented in Figure 10 illustrates the effect of varying parameters, with the labels representing ($P : N : T$). For simplicity only two dimensions are analysed, with the last figure showing the complete dataset. Given a base analysis of (4:1:10) these images show the effect of increasing P or N and decreasing T.

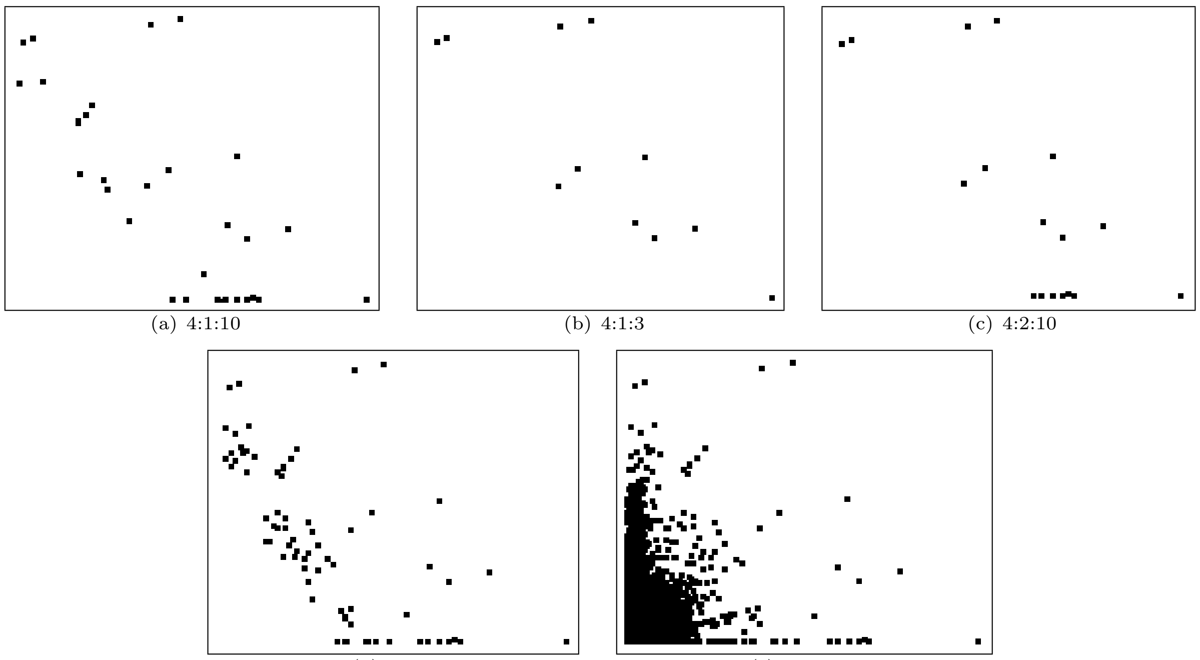
In summary, CURIO is novel and efficient, providing an outlier detection algorithm for large disk resident datasets that has linear time complexity with respect to D and efficient for $\kappa < 10$. Compared to other fast outlier detection algorithms, CURIO compares favorably against DB-outliers and GridLOF, and while FastOut is faster it is also less accurate. aLOCI is more competitive, effectively providing near linear complexity in κ. However aLOCI requires complex data structures and analysis in order to maintain this complexity, maintaining a larger footprint than CURIO. Theoretically CURIO is significantly simpler than aLOCI, with respect to both data structures and analysis, and provides an attractive alternative to aLOCI for $\kappa < 10$. This is given further weight by recent cognitive research (Pfitzner et al. 2003) which shows a significant correlation between dimensionality and the effort required by the user to understand the results. This supports our current experience, with practitioners finding it difficult to understand outlier causality when too many dimensions have been incorporated within analysis.

CURIO has been developed as part of an analysis toolkit for hospital administration data (Ceglar et al. 2006). Within this domain CURIO has proven effective at identifying outlier clusters within large complex datasets. The current application prototype is written in Java 1.4 for compatibility purposes and is multi-threaded to allow for multiple concurrent analysis instances. The tool, presented in Figure 11, is feature rich, providing clustering and characterisation in addition to outlier detection. Outlier analysis presentation is comprised of three aspects, a rich dynamically constructed scatterplot presentation and two tabular presentations providing a summary of discovered outlier clusters and more detailed view.

The CURIO prototype (currently in beta test commercial phase) is proving an effective outlier detection tool with positive feedback from test sites, especially with respect to its speed and outlier clustering ability. Feedback regarding the incorporated parameters is also positive, with users being able to understand their use within analysis.

6 Conclusion and further work

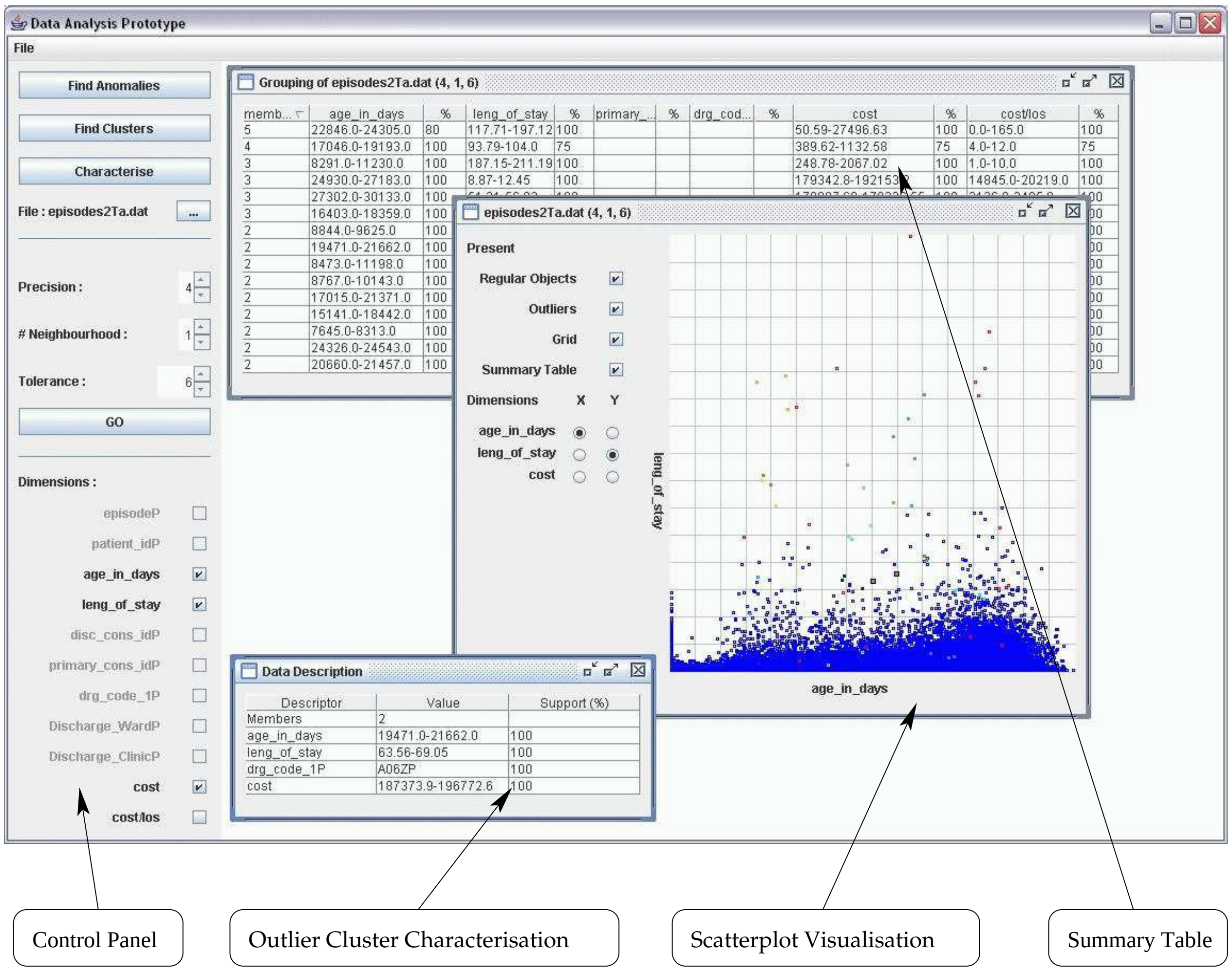
CURIO presents a novel outlier detection algorithm that makes two significant contributions. The first is the use of significant attribute bits to quantise the domain space. This technique is faster than best current practice, enabling both direct index construction from the attribute and also a single step quantisation and allocation, removing the need for intermediary mappings in both cases. The second contribution is the explicit identification of outlier clusters, which has previously been identified as an area of further research (Knorr 2002) and is proving of practical use. Furthermore, CURIO is able to analyse disk resident datasets in two sequential scans by removing object level comparisons and only storing cell counts. How-



(a) 4:1:10 (b) 4:1:3 (c) 4:2:10

(d) 5:1:10 (e) complete

Figure 10: Outlier analysis parameter variance



File : episodes2Ta.dat

Precision : 4

Neighbourhood : 1

Tolerance : 6

Dimensions :

- episodeP ☐
- patient_idP ☐
- age_in_days ✓
- leng_of_stay ✓
- disc_cons_idP ☐
- primary_cons_idP ☐
- drg_code_1P ☐
- Discharge_WardP ☐
- Discharge_ClinicP ☐
- cost ✓
- cost/los ☐

Grouping of episodes2Ta.dat (4, 1, 6)

memb...	age_in_days	%	leng_of_stay	%	primary_...	%	drg_cod...	%	cost	%	cost/los	%
5	22846.0-24305.0	80	117.71-197.12	100					50.59-27496.63	100	0.0-165.0	100
4	17046.0-19193.0	100	93.79-104.0	75					389.62-1132.58	75	4.0-12.0	75
3	8291.0-11230.0	100	187.15-211.19	100					248.78-2067.02	100	1.0-10.0	100
3	24930.0-27183.0	100	8.87-12.45	100					179342.8-192153[illegible]	100	14845.0-20219.0	100
3	27302.0-30133.0	100	[illegible]	[illegible]					[illegible]	[illegible]	[illegible]	[illegible]
3	16403.0-18359.0	100										[illegible]
2	8844.0-9625.0	100										[illegible]
2	19471.0-21662.0	100										[illegible]
2	8473.0-11198.0	100										[illegible]
2	8767.0-10143.0	100										[illegible]
2	17015.0-21371.0	100										[illegible]
2	15141.0-18442.0	100										[illegible]
2	7645.0-8313.0	100										[illegible]
2	24326.0-24543.0	100										[illegible]
2	20660.0-21457.0	100										[illegible]

episodes2Ta.dat (4, 1, 6)

Present

- Regular Objects ✓
- Outliers ✓
- Grid ✓
- Summary Table ✓

Dimensions	X	Y
age_in_days	●	○
leng_of_stay	○	●
cost	○	○

Data Description

Descriptor	Value	Support (%)
Members	2	
age_in_days	19471.0-21662.0	100
leng_of_stay	63.56-69.05	100
drg_code_1P	A06ZP	100
cost	187373.9-196772.6	100

Figure 11: Prototype Application Snapshot

ever, aside from this, CURIO's contribution lies in its simplicity, being relatively easy to understand and implement, making it an attractive alternative to current algorithms.

Three areas of further work have been identified: weighting, domain knowledge inclusion and projected analysis. Increasing the precision of particular attributes shows promise in increasing their weight within analysis, due to the positive correlation between precision and number of outliers. The inclusion of domain knowledge will improve result quality by allowing the user to identify outlier characteristics that although statistically correct, are already known and therefore removed from consideration. Projected outlier detection will allow analysis to be iteratively undertaken upon attribute subsets, facilitating processing and interpretation in a similar manner to that proposed by Aggarwal & Yu (2001).

References

Aggarwal, C. C. & Yu, P. S. (2001), Outlier detection for high dimensional data, *in* 'ACM SIGMOD International Conference on Management of Data', ACM Press, Santa Barbara, CA, USA, pp. 37–46.

Angiulli, F. & Pizzuti, C. (2002), Fast outlier detection in high dimensional spaces, *in* T. Elomaa, H. Mannila & H. Toivonen, eds, '6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2002)', Vol. 2431 of *LNCS*, Springer, Helsinki, Finland, pp. 15–26.

Arning, A., Agrawal, R. & Raghavan, P. (1996), A linear method for deviation detection in large databases, *in* 'Second International Conference on Knowledge Discovery and Data Mining (KDD'02)', ACM Press, Portland, Oregan, pp. 164–169.

Barnett, V. & Lewis, T. (1994), *Outliers in Statistical Data*, John Wiley and Sons, Chichester, New York.

Bay, S. D. & Schwabacher, M. (2003), Mining distance-based outliers in near linear time with randomization and a simple pruning rule, *in* L. Getoor, T. Senator, P. Domingos & C. Faloutsos, eds, 'Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', ACM Press, Washington, DC, USA, pp. 29–38.

Bentley, J. L. (1975), 'Multidimensional binary search trees used for associative searching', *Communications of the ACM* **18**(9), 509–517.

Billor, N., Hadi, A. S. & Velleman, P. F. (2000), 'BACON: Blocked adaptive computationally-efficient outlier nominators', *Computational Statistics and Data Analysis* **34**, 279–298.

Bolton, R. J. & Hand, D. J. (2001), Unsupervised profiling methods for fraud detection, *in* 'Credit Scoring and Credit Control VII', Edinburgh, UK.

Breunig, M., Kriegel, H., Ng, R. & Sander, J. (2000), Identifying density-based local outliers, *in* W. Chen, J. Naughton & P. Bernstein, eds, 'ACM SIGMOD International Conference on the Management of Data (SIGMOD 2000)', ACM, Dallas, TX, USA, pp. 93–104.

Ceglar, A., Morrall, R. & Roddick, J. F. (2006), Mining medical administrative data - the PKB system, *in* M. Ackermann, C. Soares & B. Guidemann, eds, 'ECML PKDD 2006 Workshop on Practical Data Mining: Applications, Experiences and Challenges', Berlin, pp. 59–66.

Chaudhary, A., Szalay, A. S. & Moore, A. W. (2002), Very fast outlier detection in large multidimensional data sets, *in* V. Ganti, ed., 'ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD)', ACM Press, Madison, Wisconsin, USA, pp. 45–52.

Chiu, A. L. M. & Fu, A. W. C. (2003), Enhancements on local outlier detection, *in* '7th International Database Engineering and Application Symposium (IDEAS2003)', ACM Press, Hong Kong S.A.R., China, pp. 298–307.

Ester, M., Kriegel, H.-P. & Xu, X. (1995), A database interface for clustering in large spatial databases, *in* '1st International Conference on Knowledge Discovery and Data Mining, KDD'95', AAAI Press, Montreal, Canada, pp. 94–99.

Ghoting, A., Parthasarathy, S. & Otey, M. (2006), Fast mining of distance-based outliers in high-dimensional datasets, *in* '2006 SIAM Conference on Data Mining', SIAM, Bethesda, MA, USA.

Hawkins, S., He, H., Williams, G. J. & Baxter, R. A. (2002), Outlier detection using replicator neural networks, *in* Y. Kambayashi, W. Winiwarter & M. Arikawa, eds, '4th International Conference on Data Warehousing and Knowledge Discovery (DaWaK '02)', Vol. 2454 of *LNCS*, Springer, Aix-en-Provence, France, pp. 170–180.

Hettich, S. & Bay, S. D. (1999), 'The UCI KDD archive [http://kdd.ics.uci.edu]: Internet usage data'.

Hodge, V. J. & Austin, J. (2004), 'A survey of outlier detection methodologies', *Artificial Intelligence Review* **22**(2), 85–126.

Jain, A. K., Murty, M. & Flynn, P. (1999), 'Data clustering: A review', *ACM Computing Surveys* **31**(3), 264–323.

Jin, W., Tung, A. K. H. & Han, J. (2001), Mining top-n local outliers in large databases, *in* '7th ACM SIGKDD International Conference on Knowledge Discovery (KDD'01)', ACM Press, San Francisco, CA, pp. 293–298.

Knorr, E. (2002), Outliers and Data Mining: Finding Exceptions in Data, PhD thesis, University of British Columbia.

Knorr, E. M. & Ng, R. T. (1998), Algorithms for mining distance-based outliers in large datasets, *in* A. Gupta, O. Shmueli & J. Widom, eds, '24th International Conference on Very Large Data Bases, VLDB'98', Morgan Kaufmann, New York, NY, USA, pp. 392–403.

Lee, W. & Xiang:, D. (2001), Information yheoretic measures for anomaly detection, *in* '2001 IEEE Symposium on Security and Privacy', IEEE Computer Society, Oakland, CA, USA, pp. 130–143.

Mahalanobis, P., Majumda, D. & Rao, C. (1949), 'Anthropometric survey of the united provinces: a statistical study', *Sankhya* **9**, 89–324.

Markou, M. & Singh, S. (2003*a*), 'Novelty detection: a review - part 1: statistical approaches', *Signal Processing* **83**(12), 2481–2497.

Markou, M. & Singh, S. (2003*b*), 'Novelty detection: a review - part 2: neural network based approaches', *Signal Processing* **83**(12), 2499–2521.

Nairac, A., Townsend, N., Carr, R., King, S., Cowley, P. & Tarassenko, T. (1999), 'A system for the analysis of jet system vibration data', *Integrated Computer-Aided Engineering* **6**(1), 53–65.

Ng, R. & Han, J. (1994), Efficient and effective clustering methods for spatial data mining, *in* J. Bocca, M. Jarke & C. Zaniolo, eds, '20th International Conference on Very Large Data Bases, VLDB'94', Morgan Kaufmann, Santiago, Chile, pp. 144–155.

Papadimitriou, S., Kitagawa, H., Gibbons, P. & Faloutsos, C. (2003), Loci: Fast outlier detection using the local correlation integral, *in* U. Dayal, K. Ramamritham & T. Vijayaraman, eds, '19th International Conference on Data Engineering (ICDE)', IEEE Computer Society, Bangalore, India, pp. 315–326.

Pfitzner, D., Hobbs, V. & Powers, D. M. (2003), A unified taxonomic framework for information visualization, *in* T. Pattison & B. Thomas, eds, 'Australian Symposium on Information Visualisation, (invis.au'03)', Vol. 24 of *CRPIT*, ACS, Adelaide, Australia, pp. 57–66.

Ramaswamy, S., Rastogi, R. & Kyuseok, S. (2000), Efficient algorithms for mining outliers from large data sets, *in* W. Chen, J. Naughton & P. Bernstein, eds, 'ACM SIGMOD International Conference on the Management of Data (SIGMOD 2000)', ACM Press, Dallas, TX, USA, pp. 427–438.

Roddick, J. F., Fule, P. & Graco, W. J. (2003), 'Exploratory medical knowledge discovery : Experiences and issues', *SigKDD Explorations* **5**(1), 94–99.

Rousseeuw, P. & Leroy, A. (1996), *Robust Regression and Outlier Detection*, 3rd edn, John Wiley and Sons.

Shekhar, S., Lu, C.-T. & Zhang, P. (2001), Detecting graph-based spatial outliers: Algorithms and applications (a summary of results), *in* '7th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD-2001)', ACM Press, San Francisco, CA, pp. 371–376.

Skalak, D. & Rissland, E. (1990), Inductive learning in a mixed paradigm setting, *in* '8th National Conference on Artificial Intelligence', AAAI Press / MIT Press, Boston, MA, pp. 840–847.

Struzik, Z. R. & Siebes, A. P. J. M. (2002), 'Wavelet transform based multifractal formalism in outlier detection and localisation for financial time series', *Physica A* **309**(3-4), 388–402.

Tao, Y., Xiao, X. & Zhou, S. (2006), Mining distance-based outliers from large databases in any metric space, *in* '12th ACM SIGKDD international conference on Knowledge discovery and data mining', ACM Press, Philadelphia, PA, USA, pp. 394–403.

Yu, D., Sheikholeslami, G. & Zhang., A. (2002), 'Findout: Finding outliers in very large datasets', *Knowledge and Information Systems* **4**(4), 387–412.

Zhang, T., Ramakrishnan, R. & Livny, M. (1996), BIRCH: An efficient clustering method for very large data bases, *in* 'ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery', ACM, Montreal, Canada, pp. 103–114.