Sleeved Co-clustering of Lagged Data

Eran Shaham[1], David Sarne[1] and Boaz Ben-Moshe[2]

[1]Department of Computer Science, Bar-Ilan University, Ramat-Gan, 52900 Israel
[2]Department of Computer Science, Ariel University Center, Ariel, 44837 Israel
Email: erans@macs.biu.ac.il, sarned@macs.biu.ac.il, benmo@ariel.ac.il

Abstract.

The paper focuses on mining clusters that are characterized by a lagged relationship between the data objects. We call such clusters ***lagged co-clusters***. A lagged co-cluster of a matrix is a submatrix determined by a subset of rows and their corresponding lag over a subset of columns. Extracting such subsets may reveal an underlying governing regulatory mechanism. Such a regulatory mechanism is quite common in real life settings. It appears in a variety of fields: meteorology, seismic activity, stock market behavior, neuronal brain activity, river flow and navigation, are but a limited list of examples. Mining such lagged co-clusters not only helps in understanding the relationship between objects in the domain, but assists in forecasting their future behavior. For most interesting variants of this problem, finding an optimal lagged co-cluster is NP-complete problem. We present a polynomial-time Monte-Carlo algorithm for mining lagged co-clusters. We prove that, with fixed probability, the algorithm mines a lagged co-cluster which encompasses the optimal lagged co-cluster by a maximum 2 ratio columns overhead and completely no rows overhead. Moreover, the algorithm handles noise, anti-correlations, missing values, and overlapping patterns. The algorithm is extensively evaluated using both artificial and real-world test environments. The first enable the evaluation of specific, isolated properties of the algorithm. The latter (river flow and topographic data), enable the evaluation of the algorithm to efficiently mine relevant and coherent lagged co-clusters in environments that are temporal, i.e., time reading data, and non-temporal.

Keywords: Clustering; co-clustering; lagged clustering; time-lagged; data mining

1. Introduction

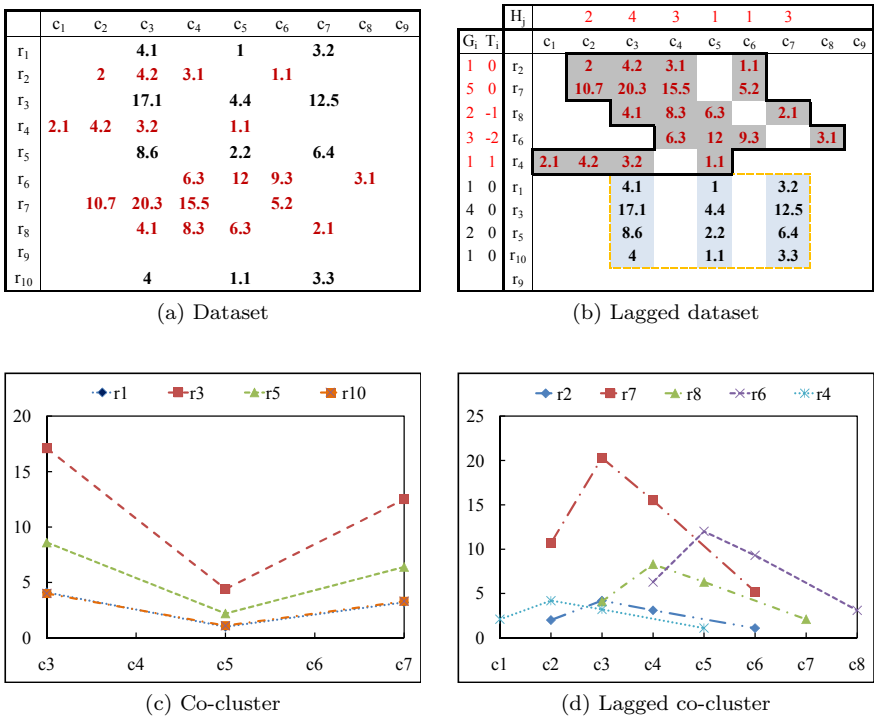
In order to benefit from the continuous improvements in digital data collection capabilities, efficient data mining and analysis tools are required. One important tool in this context, which has numerous applications is clustering (Jain, Murty and Flynn, 1999). Following seminal work by Cheng and Church (Cheng and Church, 2000) in the area of gene expression using microarray technology,

substantial focus has been placed in recent years on co-clustering (Madeira and Oliveira, 2004; Jiang, Tang and Zhang, 2004). Co-clustering extends clustering by allowing simultaneous clustering of the rows and columns of a data matrix, aiming to identify a subset of rows which exhibit similar behavior across a subset of columns, or vice versa. Many co-clustering techniques have been proposed over the years (kernel based, exhaustive enumeration, spectral analysis, greedy, CTWC, bayesian networks and others). Yet, only few have considered the problem of finding co-clusters involving lagged correlations between the behavior of a subset of rows (objects) over a subset of columns (Yin, Zhao, Zhang and Wang, 2007; Wang, Yin, Zhao and Mao, 2010). This latter case, which may reveal an underlying regulatory mechanism governing the value of the participating objects, is of great importance as it is quite common in real life settings. For example, consider the problem of identifying a group of people coordinating their movements in a crowd (e.g., trying to get from point A to point B). If the group keeps its original formation, then the trajectories of the members' spatial positions over time form a ***lagged pattern***. Similarly, consider the application of oil and gas exploration based on reflection seismology (Yilmaz and Doherty, 2001). Here, seismometers are placed on the surface, recording seismic waves. A single initiated explosion creates a wave which is reflected from each underground layer with varying time differences (depending on the depth and structure of that layer). Therefore, an appropriate time lagged analysis of the reflections received by different seismometers placed in different locations on the ground may reveal the structures and dimensions of the layers (Yilmaz and Doherty, 2001).

We denote this problem of extending co-clusters to capture lagged correlations between a subset of rows over a subset of columns as a 'lagged pattern' (see Figure 1, based on (Yin et al., 2007)). While the idea of finding lagged patterns between different streams of data is not new, existing methods are inherently limited to comparing *pairs* of objects (Kenett, Shapira and Ben-Jacob, 2009; Granger, 1969) or mining clusters consisting of *contiguous* columns (Ji and Tan, 2005; Madeira, Gonçalves and Oliveira, 2007; Huang, 2006) and thus cannot be successfully applied to the general lagged co-clustering problem.

As in most clustering problems, there are various measures for the quality of clusters found (e.g., perimeter, area). Given the fact that co-clustering is a specific case of lagged co-clustering, with a zero lag, the problem is NP-complete for any measure for which the non-lagged co-clustering problem is NP-complete. In particular, we base our model on (Cheng and Church, 2000; Madeira et al., 2007), a variant proved to be NP-complete, adding to it a lag aspect, which is also proved to be NP-complete.

The main contribution of this paper is a polynomial-time Monte-Carlo algorithm for mining *lagged co-clusters*, denoted **LC**. To the best of our knowledge, this is the first attempt to develop a polynomial approximation to the problem. The **LC** algorithm takes as input a real number matrix and a maximum error value and outputs lagged co-clusters whose errors do not exceed a 2 ratio of the pre-specified value. As part of the analysis, we prove that with fixed probability the algorithm mines a lagged co-cluster which encompasses the optimal lagged co-cluster with a maximum 2 ratio columns overhead and completely no rows overhead. This guarantee holds for any monotonically increasing objective function defined over the cluster dimensions. The **LC** algorithm handles many of the inherent shortcomings common to non-lagged data (Tanay, Sharan and Shamir, 2005). For example, it overcomes noise (erroneous reading due to local noise, equipment accuracy, experimental or human error), anti-



(a) Dataset (b) Lagged dataset

(c) Co-cluster (d) Lagged co-cluster

Fig. 1. Example dataset: (1a) example of matrix dataset. For simplicity, certain cells have been left blank in the table. (1b) the same matrix after row permutation. Two clusters emerge from (1b): A decadent type of lagged co-cluster, with no lag, located at the lower part of the matrix, marked with black fonts, blue shadow and orange dashed envelope, (visually presented in (1c)); and lagged co-cluster, located at upper part of the matrix, marked with red fonts, gray shadow and black solid envelope (visually presented in (1d)).

correlations (down-regulated, adapting gene expression terminology), missing values (e.g., due to equipment malfunction) and overlapping patterns. Furthermore, the lagged co-clusters are mined even if the amplitude of the reflected values fades along columns (as in the seismometers example). The output's structure is application dependent (e.g., maximum cluster, minimum coverage). Therefore, it is up to the application to decide how to utilize the **LC** algorithm Monte-Carlo nature of mining independent lagged co-clusters (i.e., collect into a list, set, or just, on-line processing).

The algorithm and its properties are extensively evaluated using artificial data and real-world data from two different domains (topographic data and river flow data). The artificial data is used mainly to demonstrate the efficiency of the algorithm in mining relevant and coherent lagged co-clusters, to verify the theoretical bounds and to show actual performance (e.g., run-time, hit rate). The data from the two other domains are used to demonstrate the ability of the algorithm to produce relevant and valid clusters based on overlapping, partially missing and noisy real data.

The remainder of the paper is organized as follows: in the following section we review related work. In Section 3 we formally introduce the model and show that for most interesting variants of the problem it is NP-complete. Section 4 presents the algorithm while Section 5 gives a proof of the probabilistic guarantee to efficiently mine relevant lagged co-clusters. Section 6 analyzes the running time. Section 7 presents an extension to the model while Section 8 presents the experiments conducted and their results. We conclude with a discussion and directions for future research in Section 9.

2. Related Work

A wealth of research has been undertaken studying clustering (see a survey by (Jain et al., 1999)), emerging from a variety of fields: biology, physics, economics, computer science and more. A typical clustering problem considers the case of extracting clusters from a matrix dataset where the rows represent objects and the columns represent the features of the object (Jain et al., 1999; Jiang, Tang and Zhang, 2004).

Simple mining techniques look for fully dimensional clusters: subsets of rows over *all* columns, or subsets of columns over *all* rows (Kenett et al., 2009; Erdal, Ozturk, Armbruster, Ferhatosmanoglu and Ray, 2004, inter alia). These techniques have several inherent vulnerabilities, e.g., difficulty in handling the common presence of irrelevant, noisy or missing features, and inaccuracy due to the 'curse of dimensionality' (Bellman, 1966; Moise, Zimek, Kroeger, Kriegel and Sander, 2009); all these may be counter-productive as they increase background noise (Jiang, Tang and Zhang, 2004; Madeira and Oliveira, 2004).

To overcome these obstacles, one has to find the relevant subspace for a particular pattern and ignore the rest, i.e., mining clusters contained in subset of rows over a subset of columns. This type of clustering is known as biclustering, co-clustering, co-regulation or simply clustering. The different approaches for co-clustering (see surveys by (Madeira and Oliveira, 2004; Tanay et al., 2005; Moise et al., 2009)), are based on various models: additive vs. multiplicity, axis alignment, rows over columns preferment, cluster scoring function, overlapping, etc.; and, algorithmic strategies: greedy (Cheng and Church, 2000; Ayadi, Elloumi and Hao, 2011), kernel based (Lonardi, Szpankowski and Yang, 2006; Procopiuc, Jones, Agarwal and Murali, 2002; Yang, Wang, Wang and Yu, 2003), exhaustive enumeration (Tanay, Sharan and Shamir, 2002), spectral analysis (Kluger, Basri, Chang and Gerstein, 2003), CTWC (Getz, Levine and Domany, 2000), bayesian networks (Barash and Friedman, 2002), etc. Substantial effort has been directed at non-lagged co-clustering of datasets with temporal nature (Jiang, Pei and Zhang, 2003; Bar-Joseph, Gifford, Jaakkola and Simon, 2002; Jiang, Pei, Ramanathan, Tang and Zhang, 2004; Moller-Levet, Klawonn, Cho, Yin and Wolkenhauer, 2005, inter alia), surveyed by (Roddick and Spiliopoulou, 2002), utilizing time as a natural ordering on the columns.

The lagged co-clustering model generalizes the co-clustering model by introducing lags (shifts) between the dataset's objects. Most algorithms that aim to mine lagged co-clusters do so by working on *pairs* of rows (Wu, Li and Chen, 2007; Zeng and Liu, 2008; Ramsey, Klemm, Zak, Kennedy, Thorsson, Li, Gilchrist, Gold, Johnson, Litvak et al., 2008; Chuang, Jen, Chen and Shieh, 2008). They differ in the correlation techniques being used: cross-correlation, normalized, Granger, Pearson, partial and others (Kenett et al., 2009; Granger, 1969; Baralis,

Bruno and Fiori, 2011, inter alia). Extending these algorithms to mine clusters of more than two rows requires combinatorial solutions (e.g., merging), which are both time consuming and heavily depend on the closeness merit function. In addition, correlated pairs do not necessarily have the transitive property (Kenett et al., 2009).

Among the few studies that have considered a lagged co-clustering model involving clusters of more than two rows, most are focused on a decadent variant where the goal is to find a subset of the rows over a **sequential** subset of the columns (Ji and Tan, 2005; Madeira et al., 2007; Huang, 2006; Xu, Lu, Tan and Tung, 2008).

One algorithmic approach for the sequential variant is to discretize the real number input matrix by transforming into a finite alphabet Σ. The resulting matrix, $\Sigma^{m \times n}$, enables the use of fast *string matching* techniques that run in a polynomial time (see ***q-cluster*** algorithm (Ji and Tan, 2005; Huang, 2006) and the ***CCC-Biclustering*** algorithm (Madeira et al., 2007)). The main drawback of this approach is the alphabet size. Since it requires data discretization, a coarse abstraction using a small alphabet may lead to greater errors and finer clusters being missed. Using large $|\Sigma|$ will have a dramatic influence on the run-time as it is exponentially dependent on $|\Sigma|$.

Another approach suggested for the sequential variant uses a dynamic programming method. It first searches for small coherent clusters to serve as building blocks. Then, it hierarchically merges them, while activating pruning methods (see $\boldsymbol{S^2D^3}$ algorithm (Xu et al., 2008)). The main drawback of this approach is an exponential run-time.

The work most relevant to our research is the ***ts-Cluster*** algorithm proposed by (Yin et al., 2007) and its recently evolvement, the ***td-Cluster*** algorithm, by Wang et al. (Wang et al., 2010). The ts-Cluster algorithm uses dynamic-programming and hierarchical-merging (with pruning) approach in order to mine lagged co-clusters. Its main drawback is the reduction to a small alphabet, Σ={up, non, down}, resulting with clusters representing trends rather than a more subtle model. This limitation was later removed by the td-Cluster algorithm that mines clusters of the type used in this paper. Still, the running time of both algorithms is exponential.

We note that all works cited here were also unable to find *substantial* previous reference to the lagged co-clustering problem and that the state-of-the art algorithms for this problem has exponential run-time (Yin et al., 2007; Wang et al., 2010).

3. Model

In order to present the lagged co-clustering model, we augment the legacy co-cluster definition (Cheng and Church, 2000) to include the lagging aspects. A lagged co-cluster of an $m \times n$ real number matrix X, is a submatrix determined by a subset I of the rows and their corresponding lags, denoted T ($|T| = |I|$), over a subset of the columns J, aligned to some extent to a lagged mechanism (see Figure 1). A lagged regulatory mechanism holds if for every two rows $i_1, i_2 \in I$ and their corresponding lags T_{i_1}, T_{i_2}, the proportion between the entries over all $j \in J$ is constant, independent of J: $X_{i_1,j+T_{i_1}} / X_{i_2,j+T_{i_2}} = C_{i_1,i_2}\ \forall j \in J$, revealing: a latent variable G_i indicating object i's regulation strength; a latent variable T_i indicating the influencing-lag of object i and a latent variable H_j

indicating the regulatory intensity in sample j (see Figure 1b).[1] Therefore, in a lagged co-cluster, we expect the submatrix elements to comply with the relation: $X_{i,j} \approx G_i H_{j+T_i}$ for all $(i,j) \in (I,J)$. A particular measure for the deviation in $X_{i,j}$ from the approximation $G_i H_{j+T_i}$ is the modification of the *relative error* criteria used for non-lagged co-clusters (Wang et al., 2010): $G_i H_{j+T_i}/X_{i,j}$. Our goal is to mine large submatrices, following a lagged regulatory mechanism, with a relative error below a certain pre-defined threshold:

$$\frac{1}{\eta} \leq \frac{G_i H_{j+T_i}}{X_{i,j}} \leq \eta, \ \forall \ i \in I, \ j \in J. \tag{1}$$

To facilitate analysis, we switch from a multiplicative model to an additive model by applying logarithm transformation, setting $A_{i,j} = \log X_{i,j}$, $R_i = \log G_i$, $C_{j+T_i} = \log H_{j+T_i}$ and $\varepsilon = \log \eta$. Therefore, our problem translates to finding R_i, T_i and C_j such that for all i, j,

$$-\varepsilon \leq (R_i + C_{j+T_i}) - A_{i,j} \leq \varepsilon.$$

Notice that for lagged-***anti***-correlations, i.e., $X_{i,j} \approx G_i/H_{j+T_i}$, one should use:

$$-\varepsilon \leq (R_i - C_{j+T_i}) - A_{i,j} \leq \varepsilon.$$

We note that other models, such as derivative or power-law, can be easily incorporated in the above formulation.

The optimal size submatrix depends on the merit function $f(|I|,|J|)$ used. We can rank a submatrix by its perimeter, $|I|+|J|$, area, $|I| \cdot |J|$, or any other trade-off between the number of rows and the number of columns. Previous work (e.g., (Procopiuc et al., 2002)) mainly handled biological datasets characterized by thousands of rows over tens of columns (Jiang, Tang and Zhang, 2004). Therefore, it was reasonable to consider a trade-off $\mu(|I|,|J|) = |I|/\psi^{|J|}, 0 < \psi < 1$, as in the case of $m \gg n$ the inclusion of an additional column is worth the exclusion of a relatively large number of rows. In contrast, lagged co-cluster datasets are often characterized by time readings. This results in hundreds or thousands of columns, or, in an on-line version, an infinite stream of columns. Therefore, any assumption regarding the relation between the number of rows and the number of columns is futile. Consequently, we allow the use of any monotonically growing objective function $\mu(|I|,|J|)$. Our problem thus turns into finding an *optimal size* submatrix with a relative error below a certain threshold.

Definition 1. The *sleeve-width* of a submatrix A, defined by a subset J of columns, a subset I of rows and their corresponding lag T, is:

$$sw_T(I,J) = 2 \min_{R,T,C} \max_{i \in I, j \in J} |A_{i,j} - R_i - C_{j+T_i}| \tag{2}$$

The notion of sleeve-width reflects the extent to which an entry i, j in the lagged co-cluster is allowed to deviate from being considered as the summation of $R_i + C_{j+T_i}$. The sleeve, therefore, represents the **minimal** "envelope" surrounding the deviating entries.

At this point, we have all we need in order to formally define a lagged co-cluster. However, we extend the model to include two additional parameters, β and γ, that allow the user to specify the minimum dimensions of the mined

[1] Based on the standard co-cluster model definition, according to which $\forall j \in J$, $X_{i_1,j}/X_{i_2,j} = C_{i_1,i_2}$ (multiplicative model (Wang et al., 2010); additive model (Cheng and Church, 2000)).

cluster: we denote by β the minimum number of the rows, expressed as a fraction of m and by γ the minimum number of the columns, expressed as a fraction of n.

Definition 2. Given $0 < \beta < 1$ and $0 < \gamma < 1$, constants independent of the matrix dimensions m, n. A *lagged co-cluster* of matrix A with a sleeve-width $w > 0$ is a triple (I, T, J), with J a subset of the columns, I a subset of the rows and T their corresponding lag, that satisfies the following:

–**Size:** The number of the rows is $2 \leq \beta m \leq |I| = |T|$ and the number of the columns is $2 \leq \gamma n \leq |J|$.
–**Sleeve-width:** $sw_T(I, J) \leq w$. i.e., for all $i \in I$ and $j \in J$ there are R_i, T_i and C_j, such that $|A_{i,j} - R_i - C_{j+T_i}| \leq w/2$. $R_i,\ i \in I$ will be called a column profile, $T_i,\ i \in I$ will be called a lagged column profile and $C_j,\ j \in J$ will be called a row profile.

Therefore, lagging and shifting row i by T_i and R_i, respectively, will place each column $j \in J$ within a sleeve-width of w surrounding the row profile. For the specific case where $T_i = 0$, we obtain a definition equivalent to the one used for non-lagged co-clustering (Melkman and Shaham, 2004).

Before analyzing the problem complexity we want to emphasize an important feature of the model: independency in the amplitude of an object's data. This feature is important for many applications where lagged co-clusters are likely to be found. For example, consider the seismometers application. A shock is an amplitude function H_j. Due to soil resistance, a shock fades with distance, resulting in amplitude at location T_i as $H_j * g(distance_i)$, where g is some decreasing function. Therefore, a seismometer i will present the shock readings as a function of $G_i H_{j+T_i}$, where $G_i = g(distance_i)$. Thus, two seismometers i_1 and i_2 with $distance_{i_1} < distance_{i_2}$ from the shock's source, will be located in a lagged co-cluster, with $T_1 < T_2$ and $G_1 > G_2$.

3.1. Problem Complexity

Several papers have shown that even the simplified problem of finding the largest co-cluster is NP-complete (Lonardi et al., 2006; Cheng and Church, 2000). In this subsection we generalize this result by presenting few inapproximability results regarding various optimization versions of the lagged co-clustering problem. We address two optimization problems: (1) Finding the largest single lagged co-cluster; and (2) Finding the minimal set of lagged co-clusters which covers all patterns in the data.

Observation 1. The lagged co-clustering problem (I, T, J) can be reduced to a non-lagged co-clustering problem (I, J) such that any hardness (or inapproximability) results for the non-lagged problem implies the same results for the lagged problem.

Proof. Any valid input for the co-clustering problem can be seen as a degenerate case of the lagged co-clustering problem ($lag = 0$). We reduce the lagged co-clustering problem (I, T, J) to a non-lagged co-clustering problem (I, J) by converting the lagged-matrix, A, of size $[m \times n]$, into a non-lagged one, A', as follows. Randomly choose a row $p \in A$. For every other row $i \in A$, create $2n$ new entries in A', each with a different lag in comparison to p (i.e., $-n \leq lag \leq n$).

Null entries resulting from such alignments are marked as missing values. The resulting non-lagged matrix A' is of size $[2nm \times 3n]$. □

Observation 2. Any unweighted graph $G(V, E)$ can be represented as an input to the lagged co-clustering problem.

Proof. A graph G ($|V| = n$) can be presented as a neighbors matrix $M(G) \in \{0,1\}^{n \times n}$: $M(G)[i][j] = M(G)[j][i] = 1$ iff there is an edge between vertex i and vertex j in G. We construct a co-clustering problem (I, J) by creating a matrix A and assigning $A_{i,j} = 0$ if $M(G)[i][j] = 1$ and $A_{i,j} = (n \cdot i + j)$ otherwise. This co-clustering correlation matrix is the same as $M(G)$ for small enough sleeve-width, $w < \frac{1}{n^2}$ (as the assigned values for $A_{i,j}$ are in the range 1 to n^2). Finally, using Observation 1 we move to the lagged co-clustering problem. □

Theorem 1. It is NP-complete to approximate the size of the largest combinatorial square co-cluster with an approximation factor of $n^{1-\epsilon}$.

Proof. For general graphs, it is NP-complete to approximate the size of the *Max-Clique* with an approximation factor of $n^{1-\epsilon}$ (Håstad, 1999; Khot, 2002). Following Observation 2, any approximation algorithm for finding the maximal combinatorial square in the lagged co-clustering problem can be used to approximate the *Max-Clique* problem with the same approximation factor. Thus, it can not be approximated with a factor of $n^{1-\epsilon}$, unless $P = NP$. □

Theorem 2. It is NP-complete to approximate the size of the minimal sequential cluster-set for the co-clustering problem within a constant factor (Max-SNP-Hard).

Proof. In this limited case, only sequential clusters may be considered, which can be seen as *geometric rectangles* (opposed to *combinatorial rectangles* clusters in the general case). It is NP-complete to approximate the minimal set of *geometric rectangles* covering a rectilinear polygon with holes with a constant factor ratio (Anil Kumar and Ramesh, 2003; Berman and DasGupta, 1997). Any rectilinear polygon with holes P ($|P| = n$) can be translated to a corresponding matrix $M(P) \in \{0,1\}^{n \times n}$ (Anil Kumar and Ramesh, 2003). Therefore, any algorithm, with a constant approximation factor to the limited co-clustering problem, can be used to approximate the *rectilinear polygon covering problem* with a constant factor approximation ratio. □

Theorem 3. It is NP-complete to approximate the minimal set of combinatorial squares (co-cluster set) with an approximation factor of $n^{1-\epsilon}$.

Proof. In General graphs it is NP-complete to approximate the *minimum clique partition* (MCP) with an approximation factor of $n^{1-\epsilon}$ (Zuckerman, 2007). Using Observation 2, any clique in G will be represented as a combinatorial square in $M(G)$. Therefore, approximating the minimal set of combinatorial squares with factor of $n^{1-\epsilon}$, enables the approximation of the *minimum clique partition* with the same approximation factor. □

4. The LC Algorithm

In this section, we present the **LC** algorithm, a polynomial time Monte-Carlo algorithm. Naturally, the design of **LC** is mostly influenced by solution concepts

introduced for non-lagged co-clustering problems (Lonardi et al., 2006; Melkman and Shaham, 2004). The input of the algorithm is: a matrix A of real numbers; a sleeve-width w; the minimum fraction of the rows β; and the minimum fraction of the columns γ. The algorithm outputs lagged co-clusters that comply with the specified w, β and γ. We show that, with *fixed probability*, the output contains a lagged co-cluster that encompasses the *optimal* lagged co-clusters. The mined cluster reveals the lagged rows of the optimal cluster with a maximum 2 ratio of its columns (i.e., a maximum addition of J columns) and a maximum 2 ratio of its sleeve. The algorithm makes use of random projection, which is a common technique for mining co-clusters (Lonardi et al., 2006; Procopiuc et al., 2002). It inherently handles noise and overlapping: noise by allowing the lagged co-cluster to be of some maximal pre-specified sleeve-width; overlapping by utilizing the independent random projection to reveal sub-dimensions relevant only to a specific cluster. Missing values are overcome by calculating the coherence of a lagged co-cluster on the non-missing values of the submatrix (Yang et al., 2003; Melkman and Shaham, 2004).

Figure 2 presents the **LC** algorithm. Generally, the algorithm can be divided into the following phases: (1) Initialization: randomly choose a discriminating row and a discriminating set of columns as seeds; (2) Row addition: go over all rows and lags and add those that comply with the sleeve-width criteria; (3) Column addition: go over all the columns and add the ones that comply with the sleeve-width criteria. The inclusion of a row or a column only after complying with the sleeve-width criteria, guarantees that only relevant rows and columns are added to the lagged co-cluster.

The calculation of $sw_T(I, J)$ (in lines 7, 9) is done by computing $sw(I, J)$ (Melkman and Shaham, 2004) on the non-lagged submatrix. Such a non-lagged submatrix is obtained by lagging each row $i \in I$ relative to p by T_i ($T_p = 0$). Null entries resulting from the lagging process are marked as missing values.

5. Optimality of LC Algorithm

Next we show some guarantees of the **LC** algorithm's ability to *efficiently* mine *relevant* and *coherent* lagged co-clusters. These guarantees are demonstrated experimentally in Section 8. We prove the **LC** algorithm *guarantees* finding with a fixed probability, in a polynomial number of iterations, a lagged co-cluster that encompasses an optimal lagged co-cluster. The mined lagged co-cluster will have the same lagged rows as the optimal one with a maximum 2 ratio of its columns and sleeve. The structure of the proof is inspired by (Melkman and Shaham, 2004) and consists of two major stages. First, we show that an **optimal** lagged co-cluster can be mined using a log-size discriminating set with a probability of at least 0.5. Based on this capability, we then show that when running the **LC** algorithm in a polynomial number of iterations, it mines a lagged co-cluster encompassing the **optimal** lagged co-cluster with probability of at least 0.5.

The proof relies on the important insight that a sufficient size for a discriminating set is logarithmic in the size of the set (Procopiuc et al., 2002; Lonardi et al., 2006). This latter result enables the use of a small subset, of $\mathcal{O}(\log(mn))$ size, randomly chosen from the columns, which discriminates the participating rows and their lags.

The definition of a discriminating set for the lagged model is given as follows.

LC Algorithm
Input: X, a $m \times n$ matrix of real numbers; $w \geq 0$, the maximum acceptable sleeve-width; β, the minimum fraction of rows and γ, the minimum fraction of columns.
Output: A collection of (I, T, J), each defining a lagged co-cluster based on the columns in J, the rows in I and their corresponding lags T, having a sleeve-width that does not exceed $2w$.
Initialization: Setting N and $|S|$ is thoroughly discussed in the following section.

loop N times
  randomly choose row $p: \ 1 \leq p \leq m$;
  randomly choose a set of columns S;
  set $(I, T) \leftarrow (p, 0)$;
  set $J \leftarrow S$;
  for each $(i, t): \ 1 \leq i \leq m, \ -n \leq t \leq n$ **do**
    if $sw_T(I \cup \{i\}, J) \leq w$ **then**
      add (i, t) to (I, T);
  end for
  for each $j: \ 1 \leq j \leq n$ **do**
    if $sw_T(I, J \cup \{j\}) \leq 2w$ **then**
      add j to J;
  end for
  if $|I| < \beta m$ or $|J| < \gamma n$ **then** discard (I, T, J);
end loop
return a collection of (I, T, J);

Fig. 2. **LC** algorithm for finding lagged co-clusters

Definition 3. Let (I, T, J) be a lagged co-cluster of sleeve-width w, and $p \in I$. $S \subseteq J$ is a *discriminating set* for (I, T, J) with respect to p if it satisfies:

1. $sw_T(\{i, p\}, S) \leq w$ for all $(i, t) \in (I, T)$.
2. $sw_T(\{i, p\}, S) > w$ for all $(i, t) \notin (I, T)$.

We next show that for an optimal lagged co-cluster (I^*, T^*, J^*), there are many small sets of size $\mathcal{O}(\log(mn))$, each of which is a discriminating set with a probability of at least 0.5. This latter result is important since upon selecting $p \in I^*$ and $S \subseteq J^*$, we can deduce I^*, T^* and J^*. Furthermore, the capability of finding a discriminating set with a probability of at least 0.5, is later used by Theorem 5 to mine a lagged co-cluster encompassing the optimal lagged co-cluster, in a polynomial number of iterations.

Theorem 4. Let (I^*, T^*, J^*) be an optimal lagged co-cluster of sleeve-width w, with $\gamma \leq (|J^*|/n) < \gamma'$, and let $p \in I^*$. Any randomly chosen subset S of J^*, of size $|S| \geq \log(4mn)/\log(1/3\gamma')$, is a discriminating set for (I^*, T^*, J^*), with respect to p with a probability of at least 0.5.

Proof. We show that for any S that satisfies the above, condition (1) of Definition 3 always holds and the probability that condition (2) does not hold is less than 0.5.

Let $R_i^*, \ i \in I^*$, be a column profile, $T_i^*, \ i \in I^*$ a lagged column profile,

and $C^*_j,\ j \in J^*$, a row profile for (I^*, T^*, J^*). Condition (1) of definition 3 is always satisfied, since $\{i,p\} \subseteq I^*$ and $S \subseteq J^*$, so $sw_T(\{i,p\},S) \leq sw_{T^*}(I^*,S) \leq sw_{T^*}(I^*,J^*) \leq w$.

Moving to condition (2) of Definition 3 we note that S fails to be a discriminating set for (I^*,T^*) with respect to p, only if there exists a lagged row $(i,t) \notin (I^*,T^*)$ such that $sw_T(\{i,p\},S) \leq w$. We next show that the probability of this for a particular row i and lag t is at most $(3|J^*|/n)^{|S|} < (3\gamma')^{|S|}$.

According to Definition 2, $sw_T(\{i,p\},S) \leq w$ means that there are R_i, T_i, R_p, $T_p(=0)$, and $C_j,\ j \in S$, such that:

$$|A_{i,j} - R_i - C_{j+T_i}| \leq \frac{w}{2},\ |A_{p,j} - R_p - C_{j+T_p}| \leq \frac{w}{2},\ \forall j \in S.$$

Shifting row $i \in I$ (in the first inequality) by T_i and subtracting the second inequality (of row p) we obtain $|A_{i,j} - A_{p,j} - R| \leq w$ for all $j \in S$, and some $R(= R_i - R_p)$. Due to the lagged co-cluster optimality, we show that there are no more than $3|J^*|$ columns j that satisfy this inequality.

If $|A_{i,j} - A_{p,j} - R| \leq w$ then: $-w \leq A_{i,j} - A_{p,j} - R \leq w$. After adding $(A_{p,j} - C^*_j - R^*_p)$ to both sides we obtain: $(A_{p,j} - C^*_j - R^*_p) - w \leq A_{i,j} - C^*_j - R^*_p - R \leq (A_{p,j} - C^*_j - R^*_p) + w$. Since (I^*,T^*,J^*) is an optimal lagged co-cluster, $|A_{p,j} - C^*_j - R^*_p| \leq w/2$ for all $j \in J^*$, it follows that:

$$-\frac{3}{2}w \leq A_{i,j} - C^*_j - R^*_p - R \leq \frac{3}{2}w.$$

Lemma 1. Let $J \subseteq J^*$, and let $(i,t) \notin (I^*,T^*)$. If $|A_{i,j} - C^*_j - r| \leq w/2$ for some r and all $j \in J$, then $J \subset J^*$.

Proof. The lagged co-cluster (I,T,J), with $J \supseteq J^*$, $I = I^* \cup \{i\}$ and $T = T^* \cup \{t\}$, is a lagged co-cluster of sleeve-width w satisfying $\mu(I,J) > \mu(I^*,J^*)$, contradicting the optimality of (I^*,T^*,J^*). □

Therefore, for a lagged row $(i,t) \notin (I^*,T^*)$, and for each of the intervals: $[-\frac{3}{2}w, -\frac{1}{2}w]$, $[-\frac{1}{2}w, \frac{1}{2}w]$ and $[\frac{1}{2}w, \frac{3}{2}w]$, there are at most $|J^*|$ columns j such that $A_{i,j} - C^*_j - R^*_p - r$ lies in that interval, summing to at most $3|J^*|$ columns satisfying $|A_{i,j} - A_{p,j} - R| \leq w$.

Therefore, the occurrence probability for *some* lag t $(-n \leq t \leq n)$ and *some* row i $(1 \leq i \leq m)$ is bounded (after substituting $|J^*|/n < \gamma'$ and $|S| \geq \log(4mn)/\log(1/3\gamma')$) by $2mn(3J^*/n)^{|S|} < 0.5$.

□

According to Theorem 4, any set of size $|S| \geq \log(4mn)/\log(1/3\gamma')$, randomly selected, is a discriminating set with probability of at least 0.5.

Corollary 1. The bound of $|S|$ is very hard to set as γ' is unknown. In Subsection 8.1.2 we show experimentally that a random subset of size $0.4\log(4mn)+2$ will do, freeing the user from specifying the γ'-trade-off.

Corollary 2. The minimum probability of the set S to discriminate is 0.5 and is regulated by its size. In Subsection 8.1.3 we show the relation between the discriminating set size and its discriminating probability. Moreover, in Subsection 8.1.5 we show experimentally that using a large enough discriminating set enables an exact mining of the optimal lagged co-cluster.

Theorem 5. Let S be a discriminating set for an optimal lagged co-cluster (I^*, T^*, J^*) of sleeve-width w. Provided $N \geq 2 \ln 2/\beta\gamma^{|S|}$, with a probability of at least 0.5, the **LC** algorithm will mine a lagged co-cluster (I,T,J) such that: $I = I^*$, $T = T^*$, $J \supseteq J^*$ and $|J| \leq 2|J^*|$. The sleeve-width of the lagged co-cluster is bounded by $sw_T(I, J) \leq 2w$.

Proof. Since $|I^*| \geq \beta m$, the probability of choosing a row that satisfies $p \in I^*$ is at least β. The probability of satisfying the columns discriminating set $S \subseteq J^*$ is at least $\gamma^{|S|}$, since $|J^*| \geq \gamma n$. Following Theorem 4, any given $S \subseteq J^*$ is a discriminating set with a probability of at least 0.5 with respect to p. Therefore, the probability that all N iterations fail to find a discriminating row p and a columns discriminating set S (using the inequality $(1 - 1/x)^x < 1/e$ for $x \geq 1$) does not exceed $(1 - 0.5\beta\gamma^{|S|})^N < 0.5$. It follows that **LC**'s chances of mining a lagged co-cluster upon a $p \in I^*$ and $S \subseteq J^*$ is at least 1/2. When such a lagged co-cluster is mined, we obtain from the discriminating property of S that $I = I^*$ and $T = T^*$.

We now show that J contains J^* and that $|J|$ is a maximum 2 factor of $|J^*|$. A column j is added to J only if (see line 9 in the **LC** algorithm 2): $sw_T(I, J \cup \{j\}) \leq 2w$. Meaning that $\forall i \in I$ and $\forall j \in J$ there exists R_i, T_i, and C_j such that: $-w \leq A_{i,j} - R_i - C_{j+T_i} \leq w$. Since $I = I^*$, $T = T^*$ and initially $J = S \subseteq J^*$, we obtain from (I^*, T^*, J^*) optimality that for each of the intervals $[-w, 0]$ and $[0, w]$, there are at most $|J^*|$ columns j satisfying $A_{i,j} - {R_i}^* - C_{j+T_i{}^*}$. Thus, J accumulates up to a maximum of $2|J^*|$ columns. We next show that $J^* \subseteq J$. For each $j \in J^*$, we obtain from the optimality of (I^*, T^*, J^*) that $|A_{i,j} - {R_i}^* - {C^*}_{j+T_i{}^*}| \leq w/2$, therefore, j will be added to J, namely $j \in J$. □

The extensive experimentation reported in Section 8.1 shows that for most cases $J = J^*$. Hence, in *practice*, the **LC** algorithm manages to mine the optimal lagged co-clusters with no columns or sleeve-width overhead.

6. Running time

The total number of iterations of the **LC** algorithm, is bounded according to Theorem 5 by $N = \mathcal{O}(1/(\beta\gamma^{|S|}))$. Using the method for calculating the sleeve-width (Melkman and Shaham, 2004), the inner for-loops running time is: $\mathcal{O}(\beta(m|S| + n\gamma)mn)$, where $|S| = \mathcal{O}(\log(mn))$ (see Theorem 4). In all, the running time is *polynomial* and *independent* of β: $\mathcal{O}((mn)^{2-\log\gamma})$, for some constant $0 < \gamma < 1$ independent of m, n. Experiments reported in Section 8 show that a good default value for $|S|$ is $0.4\log(4mn) + 2$ and that the actual run-time is $\mathcal{O}((mn)^{1-\log\gamma} \max\{m, n\})$.

7. Model Extensions

The lagged co-clustering model presented in Section 3, assumes a multiplicative (scaling) or additive (shifting) model with constant lags. Although such a model is well rooted in many fields (e.g., gene expression (Jiang, Tang and Zhang, 2004)), it can be generalized. In this section we present two important model extensions: (1) stretched lagging: a non-constant lag mining; and (2) vertical

transformation: a non-shift (non-scale) mining. As the extensions are based on the model described in Section 3, their analysis shares much of that given in Section 5. Therefore, we only give an outline of the algorithms with detailed explanations on the differences. The extensions are orthogonal to each other. Therefore, their consolidation results in a general and powerful lagged co-cluster model.

7.1. Stretched Lagging

The lagged co-clustering model assumes a constant lag. The meaning of a constant lag, in the context of a lagged co-cluster (I,T,J), is that the relative lag between two rows $i_1, i_2 \in I$, will be constant ($= T_{i_1} - T_{i_2}$) over **all** $j \in J$. The motivation for finding a non-constant lag, arises from several fields. Consider for example an ocean wave being recorded at different distances from the source. Each station, is not only recording the wave at a different time lag, but also, due to wave stretch, with a different wave spread (Mei, Stiassnie and Dick, 2005).

Previous work (see Section 2) considered a constant lag model of the form: $A_{i,j} \approx R_i + C_{j+T_i} = R_i + C_{\vartheta_i(j)+T_i}$, with $\vartheta_i(j) = j$. We extend the model to some general, **user given**, stretch functions $\vartheta_i(j)$. For example, $\vartheta_i(j) = 2j$ means that the lagged (T_i) signal is stretched by a factor of 2.

In order to do so, we need to modify the sleeve-width calculation method. In the regular model, a sleeve-width of $sw_T(I,J) \leq w$ means the existence of some R_i, T_i and C_j such that $-w \leq A_{i,j} - R_i - C_{j+T_i} \leq w$, for all $i \in I$ and $j \in J$. This can be represented as a system of $2|I||J|$ inequalities of $|I| + |J|$ variables. Thus, given the user function $\vartheta_i(j)$, one needs to solve a system of inequalities of the form: $-w \leq A_{i,j} - R_i - C_{\vartheta_i(j)+T_i} \leq w$. To solve such a system, one can use any linear programming (Dantzig, 1998) or CSP (Kumar, 1992) method.

7.2. Vertical Extension

The regular multiplicative model assumes (according to 1):

$$\frac{1}{\eta} \leq \frac{G_i H_{j+T_i}}{X_{i,j}} \leq \eta, \ \forall \ i \in I, \ j \in J.$$

This means that for each object $i \in I$, there is a **constant** factor G_i **scaling** it from the lagged (T_i) row profile H_j, $j \in J$. We extend the model to allow not only a scale by constant factor, but any **user given** transformation functions ξ_i, such that $X_{i,j-T_i} \approx G_i \xi_i(H_j)$. For example, consider the case of $\xi_i(H_j) = {H_j}^{\alpha_i}$. This popular transformation is known as the law of the 80%-20% (also, the *power-law* (Zipf, 1949)). One can find that law governing many fields, e.g., economics, social behavior, computer science and more (Faloutsos, Faloutsos and Faloutsos, 1999; Kang, Tsourakakis and Faloutsos, 2010). Another example is setting $\alpha_i = -1$, obtaining the private case of anti-correlation 3: $X_{i,j-T_i} \approx G_i / H_j$.

In the context of the non-lagged co-clustering model, the power-law transformation was investigated by (Zakov, 2007) and termed *Power CoClustering*. The heuristic presented there is a kernel based technique, searching for a cluster complying with the power-law model with no optimality guaranteed. The linear transformation model was investigated by (Xu, Lu, Tung and Wang, 2006; Xu

et al., 2008) and termed *shift-and-scale*. Their methods "grow" co-clusters from small coherent building blocks. Then, use hierarchical merging while pruning. The run-time of these methods is, however, exponential in the number of the input columns.

The **LC** algorithm 2 has two phases: (1) (row, lag) addition (see lines 6, 7); and (2) column addition (see lines 8, 9). The proposed algorithm, introduces another discriminating set: a row discriminating set, denoted P (in addition to the column discriminating set S), and it has three phases: (1) finding the lags, T_P, of the rows in the set P; (2) finding a subset of columns J such that (P, T_P, J) is a lagged co-cluster; and (3) finding lagged rows, (I,T), such that (I, T, J) is a lagged co-cluster. The addition of the first phase is imperative, so as to ensure a correct addition of columns (see following explanations).

The first phase aims at finding the lags of the rows in the row discriminating set P. It is similar to phase (1) of the **LC** algorithm, though limited to the rows of P:
1: randomly choose row p, $p \in P$;
2: **for each** $(i,t):\ i \in P,\ -n \leq t \leq n$ **do**
  if $sw_T(\{i,p\}, S) \leq w$ **then**
    add (i,t) to (I,T);
  end for
3: **if** $|I| < |P|$ **then** try again with different discriminating sets;
To calculate the sleeve-width (in the presence of ξ_i), we use the same technique described in the previous section (e.g., linear programming or CSP; see 7.1). Similar to the regular model analysis (see Proof 4), S is a discriminating set with probability of at least 0.5.

Now that we have the lagged rows of P, we can add columns fitting a lagged co-cluster. Therefore, the second phase uses P to discriminate columns being governed by the transformation function ξ_i. Similar to phase (2) of the **LC** algorithm, we add columns in the following manner:
1: initialize $J \leftarrow S$;
2: **for each** $j:\ 1 \leq j \leq n$ **do**
  if $sw_T(P, J \cup \{j\}) \leq w$ **then**
    add j to J;
  end for
Similar to Proof 4 (only using rows instead of columns), the probability that P is a discriminating set is at least 0.5. When it is a discriminating set, from the discriminating property of P we obtain that $J = J^*$.

In the third phase we add the remaining rows and their lags. This is done in a similar manner to phase (1) of the **LC** algorithm, except for using a sleeve-width of $2w$ instead of w:
1: initialize $I \leftarrow P$;
2: **for each** $(i,t):\ 1 \leq i \leq m,\ -n \leq t \leq n$ **do**
  if $sw_T(I \cup \{i\}, J) \leq 2w$ **then**
    add (i,t) to (I,T);
  end for
Similar to Proof 5, the resulting lagged co-cluster have the following properties: $J = J^*$, $I \supseteq I^*$ and $|I| \leq 2|I^*|$. The mined lagged co-cluster is of maximum $2w$ sleeve-width.

8. Experiments

The **LC** algorithm is extensively evaluated using both artificial and real-world data. The use of artificial data, which naturally enables tighter control of the experiments, is important as it facilitates the examination of specific, isolated properties of the algorithm. Complementary experiments with real-world data include river flow and topographic data. The latter experiments demonstrate the **LC** algorithm's capability in mining both temporal, i.e., time reading data, and non-temporal datasets.

8.1. Experiments with Artificial Data

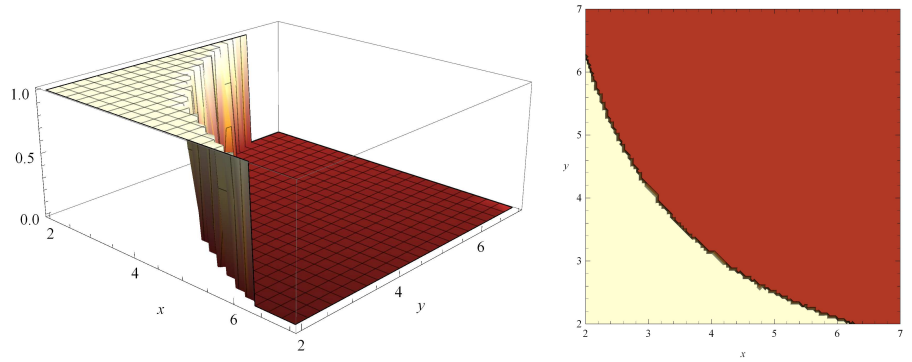
The advantage of using artificial data is that we have the maximum control in verifying the validity of clusters found (in comparison to real-world data). Specifically, the contributions of the experimentation used for the **LC** algorithm with artificial data are threefold. First, it establishes a "best practice" for the setting of parameters. Secondly, it enables the verification of theoretical bounds Finally, through the experiments, the actual run-time of the algorithm is demonstrated.[2]

8.1.1. Sleeve-width default

An artifact cluster is a submatrix that was not formed due to some hidden regulatory mechanism but as a mere aggregation of noise. Such artifacts are undesirable as they add irrelevant output. We wish to examine the lagged model from the aspect of finding artifact lagged co-clusters, i.e., whether it is common to mine such artifacts. In order to answer the question, one must specify the desired sleeve-width and the required cluster dimensions.

Given a uniformly distributed random matrix, the probability of mining an artifact lagged co-cluster (I,T,J), depends on several parameters: (1) the matrix dimensions, $[m \times n]$; (2) the lagged co-cluster dimensions, $[|I| \times |J|]$; and, (3) the sleeve-width w, $0\% \leq w \leq 100\%$. Intuitively, the larger the sleeve-width and the larger the original matrix, the greater the chance of mining artifact clusters (with an increasing probability for smaller lagged co-clusters). To examine the correlation between these parameters, we present the following upper bound probability analysis. Assume we know the column profile p. The probability of all columns $j \in J$ of a lagged row $i \in I$ to be within a sleeve-width surrounding p is: $w^{|J|}$. Thus, the probability for all rows I to form a lagged co-cluster is $w^{|I||J|}$. The probability of not having such a lagged co-cluster is therefore $1-w^{|I||J|}$. The standard representation of a lagged matrix of size $[m \times n]$ as a non-lagged one, results in a matrix of size $[2mn \times 3n]$ (see 3.1). Thus, the number of combinations to choose a set size $|I|$ out of $2mn$ rows is $\binom{2mn}{|I|}$. Similarly, choosing a set size $|J|$ out of $3n$ columns has $\binom{3n}{|J|}$ combinations. Therefore, the probability that none of the possible sub-matrices of this size in the matrix forms a lagged co-cluster is: $(1 - w^{|I||J|})^{\binom{2mn}{|I|}\binom{3n}{|J|}}$. Hence, an upper bound for the probability that at least

[2] While the number of iterations is proved to be polynomial, we want to make sure that the actual performance for large inputs is feasible.



(a) (b)

Fig. 3. Probability of an artifact lagged co-cluster in a matrix of $[1000 \times 1000]$ with sleeve-width of 5%. The x-coordinate is $|I|$ while the y-coordinate is $|J|$. Figure 3a presents a 3D view of the probability. The interesting fact is the "cliff-edge" probabilities, falling from 1.0 to 0.0. Figure 3b presents a top view of 3a. We can see the lagged co-cluster's area governing the probability curve change from 1.0 to 0.0. From both figures, we can see that a lagged co-cluster of size greater than 0.0012% of the matrix size has a probability of 0.0 to randomly appear.

one such lagged co-cluster exists is:

$$1 - (1 - w^{|I||J|})^{\binom{2mn}{|I|}\binom{3n}{|J|}}$$

To simplify understanding of the formula, we present the following charts (generated using Wolfram|Alpha (*Wolfram|Alpha*, access Dec 31, 2010)). We report here only on results with a matrix of size 1000×1000, as the results for the other matrix sizes were insignificantly different.

Figure 3a and 3b present the probability of having an artifact lagged co-cluster of sleeve-width 5% as a function of its size. The conclusion from Figure 3 is that the lagged co-cluster's size governs the probability of its random appearance.

To emphasize that fact, we present Figure 4 which depicts the relationship between the sleeve-width and the lagged co-cluster's size ($|I| = |J|$). Figure 4 reveals several insights: (1) the larger the sleeve-width the greater the probability of a larger artifact; (2) practically speaking, even for high sleeve-width, the probability of a relevant lagged co-cluster artifact is negligible. Take for example a sleeve-width of 50%. Lagged co-clusters of size greater than 60, which is 0.006% of the matrix size, have a probability of 0.0 to appear.

To conclude, lagged co-clusters of small dimensions (less than 0.006% of the matrix size) have an insignificant probability to be noise. Thus, lagged co-clusters representing a regulatory mechanism, which are naturally large in dimensions, have an insignificant probability to be noise. Therefore ordinary mining using practical dimensions, has an insignificant probability of mining artifacts.

It is notable that the sleeve-width highly depends on the "nature" of the dataset being mined. A sleeve-width of 5% of the matrix range, has been shown (Cheng and Church, 2000; Melkman and Shaham, 2004) to be a good trade-off for a *non-lagged* model, levelling between mining relevant *co-clusters* and not

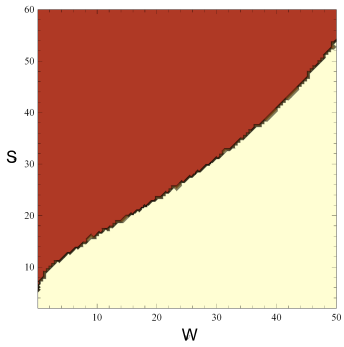


Fig. 4. Sleeve-width w vs. lagged co-cluster size s. The lower area (light pink color) represents probability of 1.0%. The upper area (red color) represents probability of 0.0%.

having artifact, falsified co-clusters, due to noise. Thus, in the absence of any prior knowledge, a sleeve-width of 5% is a good default value to use.

8.1.2. Discriminating set size

The discriminating set size, $|S|$, directly affects the run-time of the **LC** algorithm. Theorem 4 provides us with the following bound: $|S| \geq \log(4mn)/\log(1/3\gamma')$, where γ' specifies the ratio between the number of columns in an optimal lagged co-cluster and the number of matrix columns. The bound undesirably depends on γ', a parameter of which the user has no knowledge. In order to get a sense of what the value of $|S|$ is in practice, we conducted the following experiment. We first created random matrices of various sizes: from small ones of $[10 \times 10]$ to large ones of $[100000 \times 100]$. We set the dimensions of the cluster size to $\beta, \gamma \in \{0.1, 0.4, 0.6, 0.8\}$. Then we generated a random lagged co-cluster within the specified dimensions and put it at a random location in the matrix, overriding the existing values. Then, a subset of the lagged co-cluster columns was chosen at random $100,000$ times, and checked whether it was a discriminating set according to Definition 3. We consider a set of size $|S|$ to be discriminating, if it can successfully discriminate in all of the $100,000$ times.

Figure 5 depicts the relationship between $|S|$ and $\log(4mn)$. We observe the reconstruction of the linear relationship derived from Theorem 4. In addition, we obtain from Figure 5 an easy-to-use, γ' free, formula for setting $|S|$: $|S| = 0.36\log_2(4mn) + 2.33 \approx 0.4\log_2(4mn) + 2$.

8.1.3. Discriminating probability vs. discriminating set size

Experiment 8.1.2 considered a set of size $|S|$ to be discriminating, if it can successfully discriminate all N time ($N = 100,000$). In this experiment, we wish to explore the relationship between $|S|$ and its discriminating probability (in how many of the N times did the set actually discriminate). The experiment was conducted in the same manner as the previous one, only recording different sizes of $|S|$ and their discriminating capabilities.

Figure 6 describes the percentage of cases in which the subset $|S|$ discrim-

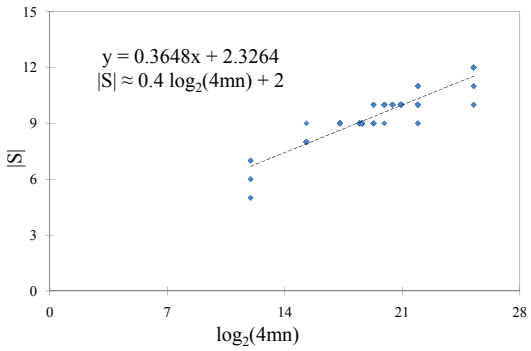


Fig. 5. Discriminating column set size $|S|$ as a function of $\log_2(4mn)$.

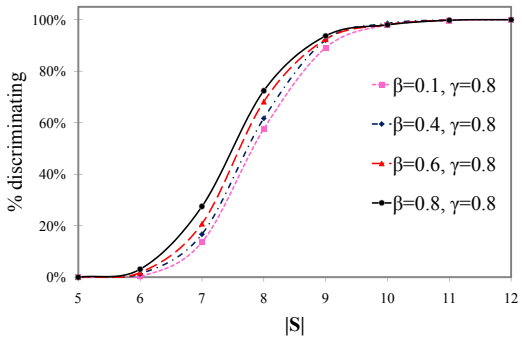


Fig. 6. The probability to discriminate as a function of the discriminating set size $|S|$.

inated. We present here only the results for $\gamma = 0.8$, as the results for other lagged co-clusters sizes, were insignificantly different.

As expected from Experiment 8.1.2, the discriminating probability for $|S| = 10$ is of $\approx 100\%$ (the matrix is of size $[1000 \times 1000]$). An important finding from the experiment, is that also for smaller sizes of $|S|$, we can have a high discriminating probability (e.g., 92% for $|S| = 9$). Since $|S|$ appears as an exponent in the estimated running time, choosing smaller $|S|$, will reduce the run-time, without having a major negative effect on the results. Figure 6 also shows also that a lagged co-cluster with fewer rows requires larger discriminating sets. The reason is that the discriminating set has to filter out more rows not belonging to the lagged co-cluster.

8.1.4. Characteristic of the discriminating set

In the previous experiments, we examined different aspects of the discriminating set. Nevertheless, in all the experiments, we choose the discriminating set, S, out of the planted lagged co-clusterś columns J^* ($S \subseteq J^*$). In this experiment, we wish to examine whether a random set of the columns, $S \subseteq n$ (not necessarily $S \subseteq J^*$), will discriminate.

The experiment was conducted in a similar manner to the previous ones. Table 1 presents different S sizes and how many of its members were contained

Table 1. Discriminating set for lagged co-clusters of $\beta = 0.8$ and $\gamma = 0.8$ in a matrix of size $[1000 \times 1000]$ with $S \subseteq n$.

$\lvert S\rvert$	$0/J^*$	$1/J^*$	$2/J^*$	$3/J^*$	$4/J^*$	$5/J^*$	$6/J^*$	$7/J^*$	$8/J^*$	$9/J^*$
2	0%	0%	0%							
3	0%	0%	0%	0%						
4	0%	0%	0%	0%	0%					
5	0%	0%	0%	0%	0%	4%				
6	0%	0%	0%	0%	0%	0%	38%			
7	0%	0%	0%	0%	0%	0%	0%	81%		
8	0%	0%	0%	0%	0%	0%	0%	0%	96%	
9	0%	0%	0%	0%	0%	0%	0%	0%	0%	99%

in J^* (i.e., $S \cap J^*$). We present here only the results for a lagged co-cluster of size $\beta = 0.8$ and $\gamma = 0.8$, planted in a matrix of size $[1000 \times 1000]$, as the results for the other matrix sizes were not significantly different.

As can be observed from table 1, a high discriminating probability is obtained for $|S| = 9$ (see last column of the last row). Such value is expected for a matrix of size $[1000 \times 1000]$ (see Experiment 8.1.2). Also, as expected from Experiment 8.1.3, the larger $|S|$, the better the discriminating probability.

The most important finding from this experiment is that S discriminates only if $S \subseteq J^*$. The explanation is that random noise is unlikely to extend a lagged co-cluster (also see Experiment 8.1.1). Therefore, only columns in J^* manage to discriminate the cluster.

The overall observation from the current experiments and the previous ones is that there is a trade off that needs to be considered when setting the size of the discriminating set S. One should choose a set large enough to discriminate (see Experiment 8.1.3) but small enough to have a high probability of being in J^* (see Experiment 8.1.4). This balance is approximated by the formula given in Experiment 8.1.2.

We wish to illuminate an interesting phenomenon related to the above experiment. When working on small lagged co-clusters, planted in a small matrix, one may get a subset $S \not\subseteq J^*$ that does discriminate. Table 2 shows the result of a lagged co-cluster of $\beta = 0.1$ and $\gamma = 0.1$, planted in a small matrix of size $[100 \times 100]$. We can see from Table 2, in contrast to Table 1, that there are, with small probability, discriminating sets of $S \not\subseteq J^*$. Two major characteristics arise: (a) The more members of S that are *not* in J^*, the greater the discriminating probability of the set; (b) The larger $|S|$, the greater its discriminating probability.

The explanation for this phenomena is rooted in the existence of small dimension random lagged co-clusters $(I_{rand}, T_{rand}, J_{rand})$. Since the lagged co-cluster is of small dimensionality, such "white noise" clusters may appear (the smaller the cluster dimensions and the greater the allowed sleeve width, see Experiment 8.1.1). The explanation for (a) is that a random cluster is unlikely to overlap with the planted lagged co-cluster. Each cluster lies in its own dimensions, having low probability of overlap with the other. Thus, the more columns of S

Table 2. Discriminating set for lagged co-clusters of $\beta = 0.1$ and $\gamma = 0.1$ in a matrix of size $[100 \times 100]$ with $S \subseteq n$.

$\|S\|$	$0/J^*$	$1/J^*$	$2/J^*$	$3/J^*$	$4/J^*$	$5/J^*$	$6/J^*$	$7/J^*$	$8/J^*$
2	0.0%	0.0%	0.0%						
3	0.1%	0.0%	0.0%	0.0%					
4	0.3%	0.2%	0.1%	0.0%	0.0%				
5	0.7%	0.4%	0.2%	0.1%	0.0%	25%			
6	1.1%	0.7%	0.4%	0.2%	0.1%	0.0%	73%		
7	1.5%	1.1%	0.7%	0.4%	0.2%	0.0%	0.0%	100%	
8	1.9%	1.5%	1.1%	0.7%	0.4%	0.2%	0.0%	0.0%	100%

that are *not* in J^* (but in J_{rand}), the better S discriminates the random lagged co-cluster. (b) The random cluster is still a valid lagged co-cluster; i.e., the larger the discriminating set the higher discriminating probability it will have.

8.1.5. Run-time, Number of Iterations and Hit Rate

Theorem 5 states that for $N \geq 2 \ln 2/(\beta\gamma^{|S|})$ tries, we are guaranteed to find a factor 2 optimal lagged co-cluster, with a probability of at least 0.5. The following experiment was conducted in order to test the practical behavior of the following boundaries: (1) The 0.5 probability boundary; (2) The number of iterations N; (3) The actual run-time it takes to find a lagged co-cluster (in ms).

For these purposes we generated a random matrix of size $[m \times n]$, $m = 1000$, $n = 1000$, with values in the range of $[100, 1100]$. Inside the matrix, a random lagged co-cluster was randomly placed, overriding the original values. The lagged co-cluster was of a random size $\beta \in [0.05 - 0.9]$, $\gamma \in \{0.3, 0.5, 0.8\}$ and a sleeve width of 5%. $|S|$ was set to 10, using the previous experimental result (see 8.1.2), for a matrix of size $[1000 \times 1000]$. Setting N to the limit given in Theorem 5, and repeating the execution of the algorithm 100 times for each cluster size, we counted: (1) Hit rate: how many times out of the 100 repetitions the algorithm managed to **precisely** mine the planted cluster; (2) Iterations: how many iterations it took in practice to mine the cluster; (3) Run-time: how long (in ms) it took to mine the cluster. The experiments were conducted using the platform: Intel core i7 (920) @ 2.67GHz CPU with 6GB RAM, Windows 7 64 bit. The algorithm was programmed in Java 1.6. The results obtained are as follows.

- **Hit Rate:** While the theoretical bound is set for 50%, the actual average hit rate is 99%. Setting $|S|$ to 10 promises such a high hit rate (see 8.1.2). Moreover, although theoretically the mined lagged co-clusters are of maximum 2 ratio of the optimal lagged co-cluster, we manage, in **all** runs, to **precisely** mine the planted lagged co-cluster, i.e., $(I, T, J) = (I^*, T^*, J^*)$. This is due to the fact that matrix's entries of random values have a very low probability in joining a lagged co-cluster.
- **Number of Iterations:** Figure 7 presents the actual number of iterations needed to mine a lagged co-cluster in relation to the theoretical boundary.

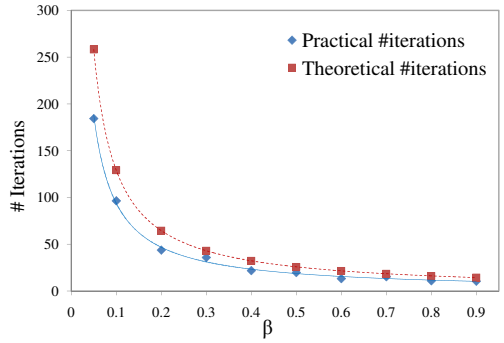


Fig. 7. Number of iterations needed to mine a lagged co-cluster vs. the theoretical bound, for $\gamma = 0.8$.

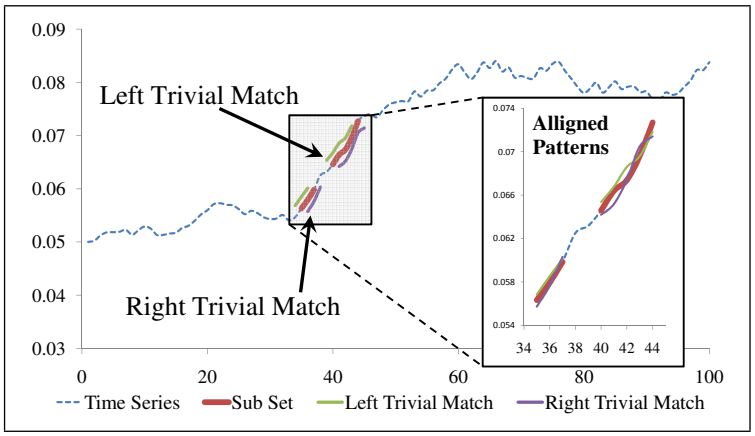


Fig. 8. A random walk illustrating the Trivial Match phenomena.

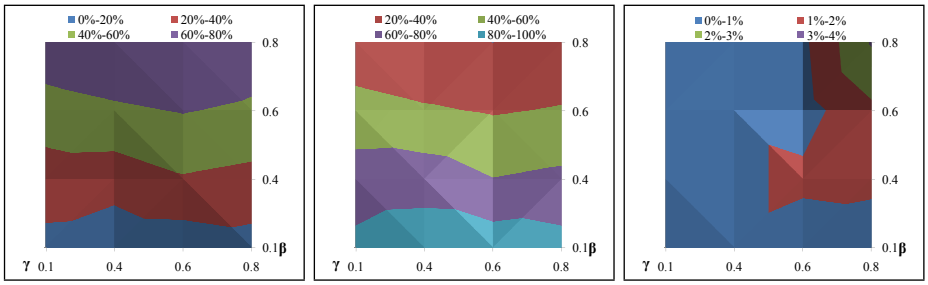
We present here only results of $\gamma = 0.8$, as the results for the other γ values were insignificantly different. Since both γ and $|S|$ were held fixed at 0.5 and 10, respectively, both theoretical and practical situations present a behavior of $N = \mathcal{O}(1/\beta)$.

– **Run-time:** The boundary specified in Section 6 is: $t = \mathcal{O}((mn)^2/\gamma^{|S|})$. Fitting the actual run-time to an equation of type: $t = c/(\beta^x \gamma^y)$ (t in ms), where $c = (mn)^2$, we obtain: $x = -0.4$, $y = 8.9$ and $c = 325$. As expected, the power of β is almost 0 and the power of γ is close to 10 ($|S| = 10$). Also, the c value indicates a dependency of $\mathcal{O}(mn \max\{m, n\})$ rather than of $\mathcal{O}((mn)^2)$.

To summarize, using the suggested discriminating set size, the **LC** algorithm manages to mine the *optimal* lagged co-clusters in our test set with a probability of 99% and do so in a feasible running time.

8.1.6. Trivial Match

Consider the random walk example illustrated in Figure 8 based on (Chiu, Keogh and Lonardi, 2003). One can observe that the subset of columns in red (the



(a) Trivial Match. (b) Non discriminating. (c) Discriminating.

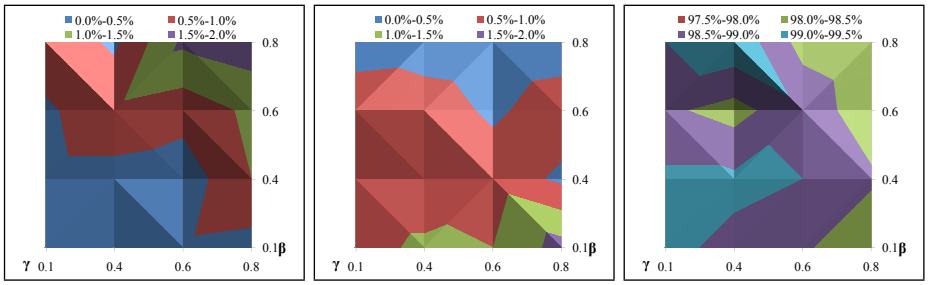
Fig. 9. Discriminating breakdown for $|S|$=6. Figure 9a presents the non-discriminating probability due to Trivial Match. Figure 9b presents the non-discriminating probability excluding the Trivial Match cases. Figure 9c presents the discriminating probability. Here, there is a high probability for non-discrimination (9a) and (9c), or, an insignificant probability to discriminate (9c), since $|S|$=6 is too small to discriminate for a matrix of size [1000 × 1000] (see Experiment 8.1.2).

middle bold line), is very close in pattern to the subset lagged immediately to the left (green) and to the right (purple). The phenomena is called *Trivial Match* and is usually characterized by matching patterns located close to the one mined. It mostly happens for smooth, slowly changing subsequences whereas rapidly changing, noisy subsequences, will have very few such trivial matches, if any. While (Chiu et al., 2003) investigate Trivial Match in the context of self patterns in a time series, it may also arise in the lagged co-clustering model. Let us consider the following special situation: our dataset contains a row that is a discrete version of the function $y = ax$. If the discriminating set S has the form of $y = ax + b$, then all time-lags will pass the discrimination criteria. Therefore, the row will appear $\mathcal{O}(n)$ times in the lagged co-cluster with different lags.

In the following experiment, we wish to examine the severity of the phenomena in a lagged co-clustering model. The experiment was conducted in the same manner as Experiment 8.1.2, on a [1000 × 1000] matrix, recording 10,000 cases where the set was **not** a discriminating one (see Definition 3), either because: (a) There exists a row $i \notin I$ such that $sw_T(\{i, p\}, S) \leq w$; or (b) There exists a row $i \in I$ such that $sw_T(\{i, p\}, S) > w$. The latter case, is the case of Trivial Match.

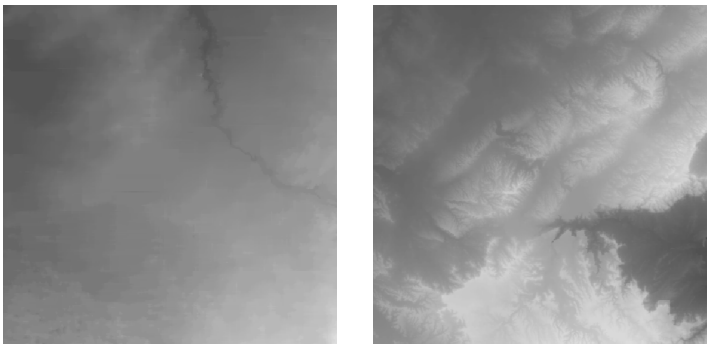
Figure 9 presents the breakdown of discriminating probability for $|S|$=6. Figure 10 presents the breakdown of discriminating probability for $|S|$=10. From Figure 9, we can see the correlated increase in both β and Trivial Match probability. Since more and more rows belong to the lagged co-cluster, the use of a small discriminating set ($|S|$=6) cause them to fail on Trivial Match. From Figure 10, we can see that the probability to discriminate is high in all ranges of β and γ and therefore, the probability for a Trivial Match is very low.

To conclude, although the phenomena of Trivial Match may result in a corrupted lagged co-cluster, it can be easily avoided using a large discriminating set. A good estimation for such size is given in Experiment 8.1.2.



(a) Trivial Match. (b) Non discriminating. (c) Discriminating.

Fig. 10. Discriminating breakdown for $|S|$=10. Figure 10a presents the non-discriminating probability due to Trivial Match. Figure 10b presents the non-discriminating probability excluding the Trivial Match cases. Figure 10c presents the discriminating probability. Here, there is a high probability to discriminate (10c), or, an insignificant probability for non-discrimination (10a) and (10c). Using $|S|$=10 is large enough to have a high discriminating probability for a matrix of size [1000 × 1000] (see Experiment 8.1.2).



(a) Smooth terrain with a stream (b) Highly complex noisy terrain

Fig. 11. Elevation maps - the brighter the greater the elevation. Each map is of size 25×25 km^2. Map (11a) represents a **simple** terrain with few dramatic changes. Map (11b) represents a **complex** terrain with many hills, mountains, canyons, etc.

8.2. Experiments on Topographic Data

A topographic map is a 3D representation of a surface (see Figure 11). Such a map is often represented as a Digital Elevation Map (DEM) which is a grid-based sample of the surface elevation. Detailed DEM maps are large datasets, e.g., consider a 100×100 km^2 map with 1 meter grid sample. Such a map has 10^{10} elevation samples - or in other words such a map has 10 Giga pixels. Manipulating and querying such datasets often requires sophisticated algorithms, and in many cases due to the nature of the problem or the size of the dataset, efficient heuristics rather than exact ones (Abraham and Roddick, 1999).

In this experiment, we wish to examine the capability of the **LC** algorithm

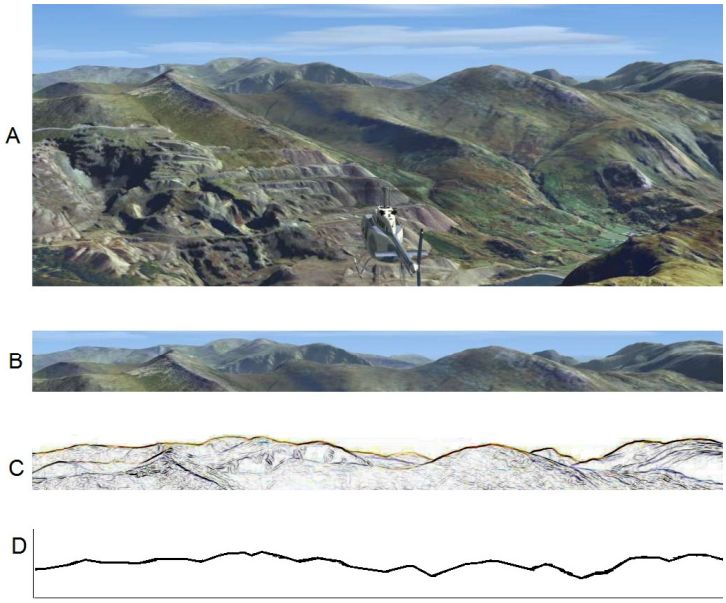


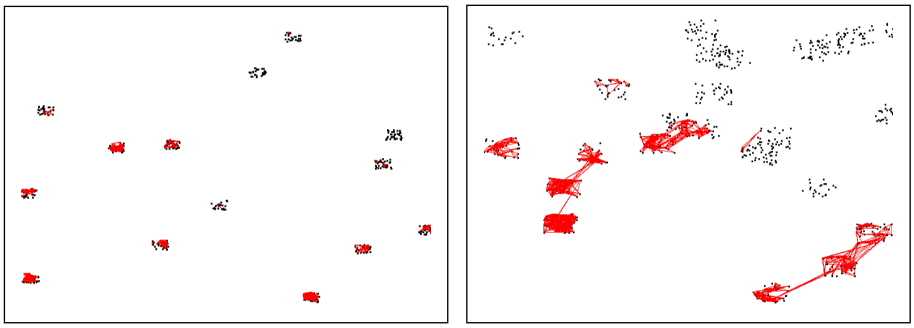
Fig. 12. Skyline computation: (A) The original figure. (B) The horizon, only. (C) Edge detection. (D) The Skyline blocking angle, X-axis represents horizontal angle, Y-axis represents the vertical angle (blocking angle).

to cluster random viewpoints, i.e., skylines seen from *different angles*. The experiment consisted of 17 different high-resolution elevation maps representing various types of terrain such as plains, hills, mountains, lakes and dunes. Each map represents a rectangular area of 100×100 km^2, and includes 1.6×10^7 grid samples (sampling every 25m). For each terrain, 100-3000 random locations were chosen within it using two steps: (1) Locating 10-50 random points as centers; (2) For each location, ascribing a random center from which the distance is within a random range of [0.5-4] km. In order to overcome minor obstacles within close range of the observer, a random height value in the range of [10-30] meters above ground was assigned to each location. In order to create a *viewpoint*, each location was assigned a random **angle** value a_0 in the range of [0,60] degrees representing the skyline starting angle. Each skyline consisted of 300 samples representing the angle range of $[a_0, a_0 + 300]$ degrees. For each angle the maximal $z - blocking$ angle was computed, as shown in Figure 12.

The results obtained indicate that the algorithm can mine precise and valid clusters. Figures (13a), (13b) and (13c) present lagged co-cluster results for various terrain maps. Each black dot represents a viewpoint, while red lines represent a lagged co-cluster, binding those viewpoints.

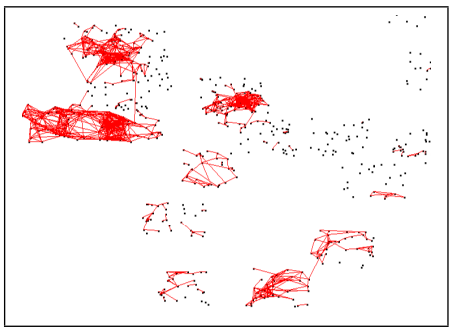
It would seem only natural for this dataset that, in every mined lagged co-cluster, the angle between any two viewpoints will be equal to the lag between them. Figure 14 presents the probability of occurrences of $(Angle - Lag)$ (0 means $Angle \equiv Lag$) corresponding to the above terrain maps. The graph shows that in all the different terrain maps, there is a very high chance of the mined *lag* being very close to the *angle* of the viewpoint.

To strengthen our belief that clusters in such datasets can only be found



(a) simple terrain map

(b) complicated terrain map



(c) complex terrain map

Fig. 13. Lagged co-clusters in different terrain types. Map (13a) represents a *simple* terrain. The terrain is characterized by tight, remotely situated centroids, with condensed surrounding viewpoints. There are no artifact clusters found (no inter-cluster line). Map (13b) represents a *complicated* terrain. The terrain is characterized by clusters' centroids being close to each other, while the clusters' diameters are large (viewpoints are spread). The mined lagged co-clusters are still relevant (belong to the same spatial cluster) and tight, as there are very few inter-cluster connections. Map (13c) represents a *complex* terrain. The terrain contains close, sometimes overlapping spatial clusters with highly scattered viewpoints. This makes it hard in some cases, to even visually classify a viewpoint to a spatial cluster. Here, we have a reasonable number of artifact clusters (inter-cluster lines).

using the lagged model, we ran a non-lagged co-cluster algorithm (Melkman and Shaham, 2004) on the same datasets, with the same clustering requirements. As expected, in this latter experiment no co-clusters were found.

In conclusion, we have shown that, given a topographic map and a skyline view, it is possible to derive the location from which the skyline was seen using the mining of lagged co-clusters. The **LC** algorithm performed well in this case, managing to successfully mine lagged co-clusters from different terrain types, with a small number of artifact clusters. The algorithm can thus be used as a classifier in this domain and even as a means of navigation if used continuously. In addition to the derived spatial location, we can also infer the angle of the

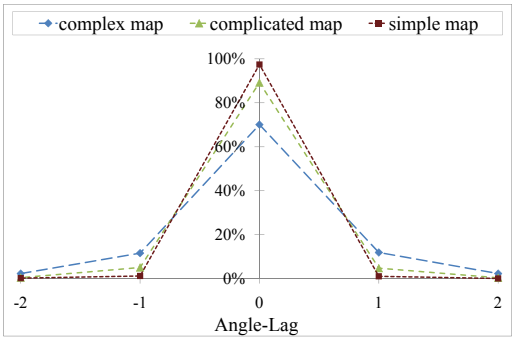


Fig. 14. Relative angle of viewpoint vs. relative lag. For the complex terrain map (13c) there is a 70% chance of the lag being equal to the angle, and in total, 94% chance for the lag to be in the range ± 1 from the angle of viewpoint. For the complicated terrain map (13b) the chance of the lag equaling the angle rises to 89% with a total chance of 99% of being in the range of ± 1. For the simple terrain map (13a) the chance of a lag equaling the angle is 97% with a total chance of almost 100% of being in the range of ± 1.

viewpoint. The failure of the non-lagged algorithm and the success of the **LC** algorithm implies that the lagged co-clusters were mined due to their lagged nature and not as a result of terrain properties. However, terrains that are very flat (i.e., no reference points) or highly noisy (a slight location change may lead to a significant skyline change due to the effect of hiding and distortions) can be a challenge.

8.3. Experiments on River Flow Data

The final dataset used for our experiments was real-time **water** data obtained from the U.S. Geological Survey (USGS) (*USGS: Real Time Water Information System*, 2010). We compiled a dataset containing flow readings of rivers in the states of New Mexico, Colorado and Nevada. There are 539 rows (objects) each representing a gauge. The columns show the gauge's discharge (ft^3/s) readings for March 2010, sampled every 15 minutes, resulting in 2877 columns.

A relevant lagged co-cluster will naturally be formed by readings from gauges located along the same river, as water flowing downstream will present a correlated flow between different measuring stations with a lag of time.

The flow of a stream depends on multiple parameters. Many of them change dramatically over time and space: nature (local and global weather conditions, joining and forking rivers, water evaporation, water loss through the river bed, etc.) and human influence (dams, factories, irrigation pools, settlements, pumping stations, sewage systems, etc.). In addition, the data is very noisy due to human and equipment inaccuracy and characterized by a high missing data ratio (**23%**) caused by various reasons: equipment malfunction, river conditions (effect of ice, flood damage, zero flow), station only recording seasonally, etc. Therefore, mining such datasets for lagged co-clusters is highly complicated.

We consider a cluster to be accurate if *all* the participating gauges are located within the same basin. We note that it is unlikely to mine a cluster containing

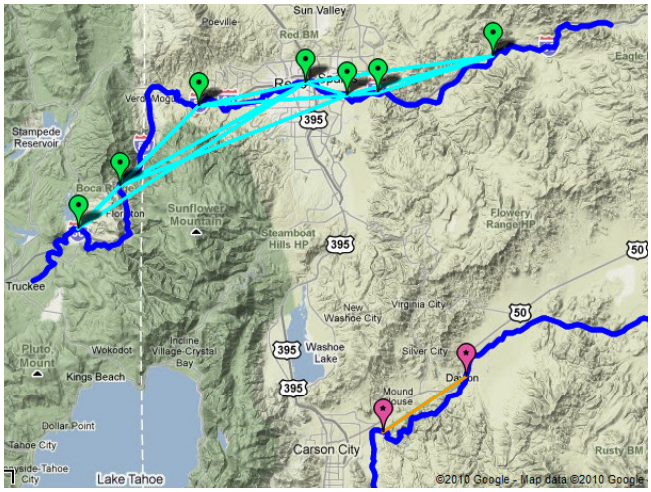


Fig. 15. Northern river - lagged co-clusters along the Pyramid river, Nevada, USA. There are 7 stations and 14 lagged co-clusters marked by dotted green balloons (gauge stations) and cyan lines (connecting cluster's members). Southern river - a lagged co-cluster along the Carson river, Nevada, USA. Pink starred balloons mark the gauges while an orange line marks connected cluster's members. For visibility purposes, the rivers' routes are partially painted in blue.

all the gauges in a basin. For example, gauges located at the exit of a dam or those which are malfunctioning would not be included.

Using the **LC** algorithm, we mined 488 lagged co-clusters (see example in Figure 15). Of those, 461 clusters (94%) were in the same state. Manually checking the 27 inter-state clusters, we found that at least one gauge is located at the exit of a dam thus changing the flow of the river. 405 clusters (83%) were in the same basin. Manually checking the 56 inter-basin clusters reveals the following reasons for the mismatch: (1) Technical administrative division of basins into upper middle and lower part (12 clusters); (2) Fork in a river: streams merging from different basins (36 clusters); (3) Gauges are located in swampy areas (8 clusters). Therefore, we achieved an accuracy of 94% on a state granularity and an accuracy of 93% on a basin granularity (considering (1) and (2) above as valid), while providing relevant explanations for the artifact lagged co-clusters (dams and swampy areas) enabling future pre-processing exclusion.

Since water flows downstream, we expect the lag difference between any two stations to be positive if their altitude difference is also positive. Out of the 488 clusters found, 92% followed the above logic. The other 8% had the following characteristics: (1) Human intervention, e.g., a dam or an irrigation area (35 clusters); (2) Environment factors, e.g., a high lake feeding two streams (6 clusters). Therefore, the lag proved to be a good indication of the direction of water flow.

As with the topographic data, we ran a non-lagged clustering algorithm (Melkman and Shaham, 2004) on the above dataset, finding only 4 clusters (in comparison to 488 found by the **LC** algorithm). All clusters were trivial and caused by: (1) Irrigation area; (2) Station position at the exit of a dam; (3)

Short distance between stations (i.e., river sampling of 15 minutes is a gross granularity).

9. Discussion, Conclusions and Future Work

The importance of co-clustering is unquestionable and has been thoroughly discussed and demonstrated in cited prior work. The lagged co-clustering model generalizes the co-clustering model, enabling the inclusion of an additional important dimension, a **lag aspect**, in the regulatory paradigm. The real-life datasets used in the former section were large, highly noisy, contained many missing values and were rich in overlapping clusters. While the **LC** algorithm managed to find precise, coherent and relevant lagged co-clusters in a practicable time and with almost no artifacts, the use of a non-lagged co-clustering method did not result in any relevant clusters. This encouraging result is important for model validation and suggests great potential for mining lagged co-clusters in many other fields of science, technology and medicine, e.g., gene expression data (Kluger et al., 2003; Getz et al., 2000), MRI data (Jain et al., 1999). It is notable that not only datasets with a time aspect can benefit from such use of the algorithm, and the lagged aspect can have various interpretations (e.g., in the topographic dataset used in our experiment, the lagged aspect is the point of view).

As a generalization of the co-clustering problem, the lagged problem is NP-complete for most interesting optimality measure. The **LC** algorithm presented in this paper relies on a strong theoretical base, enabling a probability promise on mining a near optimal lagged co-cluster and a theoretical bound to the number of iterations it will take. Unlike other algorithms, **LC** does not assume any specific scoring merit on the mined clusters. Experiments on artificial data shows that practically, the **LC** algorithm manages to mine the optimal lagged co-cluster. Since the **LC** algorithm iterations are independent, the use of parallel computing or special hardware can boost the performance even further.

The algorithm has several configurable parameters, for which this paper presents default values. As in non-lagged co-clustering models, one of the key parameters that needs to be set carefully in order to mine meaningful clusters is the sleeve-width. Setting it too high might result in many artifact clusters, while setting it too low might preclude valid clusters. In order to choose an appropriate value for this parameter, one can adopt any of the methods suggested for non-lagged co-clustering, e.g., gradual increase, starting from a relatively small sleeve.

As in non-lagged co-clustering, the ability to mine lagged co-clusters offers important functionalities, e.g., using the tool as a classifier. Nevertheless, unlike non-lagged co-clustering, the ability to mine lagged co-clusters encapsulates also a **forecasting** functionality, which can be highly useful in numerous applications ranging from meteorology to stock markets. While the current results supply some basic forecasting functionality (following the lagged-pattern found), we believe there is far more that can be developed in this aspect in terms of future research.

Acknowledgements. This work was supported in part by the Israeli Science Foundation grant no. 1401/09.

References

Abraham, T. and Roddick, J. (1999), 'Survey of spatio-temporal databases', *GeoInformatica* **3**(1), 61–99.

Anil Kumar, V. and Ramesh, H. (2003), 'Covering rectilinear polygons with axis-parallel rectangles', *SIAM journal on computing* **32**(6), 1509–1541.

Ayadi, W., Elloumi, M. and Hao, J. (2011), 'BicFinder: a biclustering algorithm for microarray data analysis', *Knowledge and Information Systems* pp. 1–18.

Bar-Joseph, Z., Gifford, D., Jaakkola, T. and Simon, I. (2002), A new approach to analyzing gene expression time series data, *in* 'Proceedings of the sixth annual international conference on Computational biology', ACM, pp. 39–48.

Baralis, E., Bruno, G. and Fiori, A. (2011), 'Measuring gene similarity by means of the classification distance', *Knowledge and Information Systems* pp. 1–21.

Barash, Y. and Friedman, N. (2002), 'Context-specific Bayesian clustering for gene expression data', *Journal of Computational Biology* **9**(2), 169–191.

Bellman, R. (1966), 'Dynamic Programming', *Science* **153**(3731), 34–37.

Berman, P. and DasGupta, B. (1997), 'Complexities of efficient solutions of rectilinear polygon cover problems', *Algorithmica* **17**(4), 331–356.

Cheng, Y. and Church, G. (2000), Biclustering of expression data, *in* 'Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology', Vol. 8, AAAI, pp. 93–103.

Chiu, B., Keogh, E. and Lonardi, S. (2003), Probabilistic discovery of time series motifs, *in* 'Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 493–498.

Chuang, C., Jen, C., Chen, C. and Shieh, G. (2008), 'A pattern recognition approach to infer time-lagged genetic interactions', *Bioinformatics* **24**(9), 1183–1190.

Dantzig, G. (1998), *Linear programming and extensions*, Princeton Univ Pr.

Erdal, S., Ozturk, O., Armbruster, D., Ferhatosmanoglu, H. and Ray, W. (2004), A time series analysis of microarray data, *in* 'Proceedings of the 4th IEEE Symposium on Bioinformatics and Bioengineering', IEEE, pp. 366–378.

Faloutsos, M., Faloutsos, P. and Faloutsos, C. (1999), On power-law relationships of the internet topology, *in* 'Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication', ACM, pp. 251–262.

Getz, G., Levine, E. and Domany, E. (2000), 'Coupled two-way clustering analysis of gene microarray data', *Proceedings of the National Academy of Sciences* **97**(22), 12079–12084.

Granger, C. (1969), 'Investigating causal relations by econometric models and cross-spectral methods', *Econometrica: Journal of the Econometric Society* **37**(3), 424–438.

Håstad, J. (1999), 'Clique is hard to approximate withinn 1- ε', *Acta Mathematica* **182**(1), 105–142.

Huang, J. (2006), Identifying co-regulated gene group from time-lagged gene cluster using cell cycle expression data, PhD thesis, National Central University, Taiwan.

Jain, A., Murty, M. and Flynn, P. (1999), 'Data clustering: a review', *ACM computing surveys* **31**(3), 264–323.

Ji, L. and Tan, K. (2005), 'Identifying time-lagged gene clusters using gene expression data', *Bioinformatics* **21**(4), 509–516.

Jiang, D., Pei, J., Ramanathan, M., Tang, C. and Zhang, A. (2004), Mining coherent gene clusters from gene-sample-time microarray data, *in* 'Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 430–439.

Jiang, D., Pei, J. and Zhang, A. (2003), Interactive exploration of coherent patterns in time-series gene expression data, *in* 'Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 565–570.

Jiang, D., Tang, C. and Zhang, A. (2004), 'Cluster analysis for gene expression data: A survey', *IEEE Transactions on Knowledge and Data Engineering* **16**(11), 1370–1386.

Kang, U., Tsourakakis, C. and Faloutsos, C. (2010), 'Pegasus: Mining peta-scale graphs', *Knowledge and Information Systems* pp. 1–23.

Kenett, D., Shapira, Y. and Ben-Jacob, E. (2009), 'RMT Assessments of the Market Latent Information Embedded in the Stocks' Raw, Normalized and Partial Correlations', *Journal of Probability and Statistics* .

Khot, S. (2002), Improved inapproximability results for maxclique, chromatic number and approximate graph coloring, *in* 'Proceedings of the 42nd IEEE symposium on Foundations of Computer Science', IEEE, pp. 600–609.

Kluger, Y., Basri, R., Chang, J. and Gerstein, M. (2003), 'Spectral biclustering of microarray data: coclustering genes and conditions', *Genome Research* **13**(4), 703–716.

Kumar, V. (1992), 'Algorithms for constraint-satisfaction problems: A survey', *AI magazine* **13**(1), 32–44.

Lonardi, S., Szpankowski, W. and Yang, Q. (2006), 'Finding biclusters by random projections', *Theoretical Computer Science* **368**(3), 217–230.

Madeira, S. C., Gonçalves, J. P. and Oliveira, A. L. (2007), Efficient Biclustering Algorithms for identifying transcriptional regulation relationships using time series gene expression data, Technical Report 22/2007, INESC-ID.

Madeira, S. and Oliveira, A. (2004), 'Biclustering algorithms for biological data analysis: a survey', *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **1**(1), 24–45.

Mei, C., Stiassnie, M. and Dick, K. (2005), *Theory and Applications of Ocean Surface Waves: Nonlinear aspects*, World Scientific.

Melkman, A. and Shaham, E. (2004), Sleeved CoClustering, *in* 'Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 635–640.

Moise, G., Zimek, A., Kroeger, P., Kriegel, H. and Sander, J. (2009), 'Subspace and projected clustering: experimental evaluation and analysis', *Knowledge and Information Systems* **21**(3), 299–326.

Moller-Levet, C., Klawonn, F., Cho, K., Yin, H. and Wolkenhauer, O. (2005), 'Clustering of unevenly sampled gene expression time-series data', *Fuzzy sets and Systems* **152**, 49–66.

Procopiuc, C., Jones, M., Agarwal, P. and Murali, T. (2002), A Monte Carlo algorithm for fast projective clustering, *in* 'Proceedings of the 2002 ACM SIGMOD international conference on Management of data', ACM, pp. 418–427.

Ramsey, S., Klemm, S., Zak, D., Kennedy, K., Thorsson, V., Li, B., Gilchrist, M., Gold, E., Johnson, C., Litvak, V. et al. (2008), 'Uncovering a macrophage transcriptional program by integrating evidence from motif scanning and expression dynamics', *PLoS Computational Biology* **4**(3).

Roddick, J. and Spiliopoulou, M. (2002), 'A survey of temporal knowledge discovery paradigms and methods', *IEEE Transactions on Knowledge and data engineering* pp. 750–767.

Tanay, A., Sharan, R. and Shamir, R. (2002), 'Discovering statistically significant biclusters in gene expression data', *Bioinformatics* **1**(1), 1–9.

Tanay, A., Sharan, R. and Shamir, R. (2005), 'Biclustering algorithms: A survey', *Handbook of computational molecular biology* **9**, 26–1.

USGS: Real Time Water Information System (2010), U.S. Geological Survey, National Water Information System. http://waterdata.usgs.gov/nwis/.

Wang, G., Yin, L., Zhao, Y. and Mao, K. (2010), 'Efficiently Mining Time-Delayed Gene Expression Patterns', *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics* **40**(2), 400–411.

Wolfram|Alpha (access Dec 31, 2010), Wolfram Alpha LLC. http://www.wolframalpha.com/.

Wu, W., Li, W. and Chen, B. (2007), 'Identifying regulatory targets of cell cycle transcription factors using gene expression and ChIP-chip data', *BMC bioinformatics* **8**(1), 188.

Xu, X., Lu, Y., Tan, K. and Tung, A. (2008), Finding Time-lagged 3D Clusters, *in* 'Proceedings of the 24th International Conference on Data Engineering', pp. 445–456.

Xu, X., Lu, Y., Tung, A. and Wang, W. (2006), Mining Shifting-and-Scaling Co-Regulation Patterns on Gene Expression Profiles, *in* 'Proceedings of the 22nd International Conference on Data Engineering', IEEE Computer Society, pp. 89–98.

Yang, J., Wang, H., Wang, W. and Yu, P. (2003), Enhanced Biclustering on Expression Data, *in* 'Proceedings of the 3rd IEEE Symposium on BioInformatics and BioEngineering', IEEE, pp. 321–327.

Yilmaz, O. and Doherty, S. (2001), *Seismic data analysis*, Society of Exploration Geophysicists.

Yin, Y., Zhao, Y., Zhang, B. and Wang, G. (2007), 'Mining Time-Shifting Co-regulation Patterns from Gene Expression Data', *Advances in Data and Web Management* pp. 62–73.

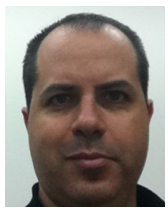
Zakov, S. (2007), Power CoClustering: A Model Guided Approach for Automated Recognition of Trascription Reguratory Mechanism by Gene Expression Data Analysis, PhD thesis, Ben Gurion University, Israel.

Zeng, T. and Liu, J. (2008), Analysis on Time-lagged Gene Clusters in Time Series Gene Expression Data, *in* 'Proceedings of the 2007 International Conference on Computational Intelligence and Security', IEEE, pp. 181–185.

Zipf, G. (1949), *Human behavior and the principle of least effort: An introduction to human ecology*, addison-wesley press.
Zuckerman, D. (2007), 'Linear degree extractors and the inapproximability of max clique and chromatic number', *Theory of Computing* **3**(1), 103–128.

Author Biographies

Eran Shaham In 1998, Eran Shaham received his B.Sc. degree in Mathematics and Computer Science from Ben-Gurion University, Israel. From 1999 to 2001, he worked at Parametric Technology Corporation (PTC), Israel. In 2004, he received his M.Sc. degree in Computer Science from Ben-Gurion University, Israel. From 2005 to 2008, he worked at the IBM Haifa Research Lab, Israel. He is currently a Ph.D. student at the Department of Computer Science, Bar-Ilan University, Israel. His research interests include data mining in general and its lagged aspects in particular.



David Sarne David Sarne is a senior lecturer in the Computer Science department in Bar-Ilan University, Israel. He received a B.Sc., M.Sc., and a Ph.D. degree in Computer Science from Bar-Ilan University, Israel. During 2005-2007 he was a post-doctoral fellow at Harvard University. His research interests include economic search theory, market mechanisms for forming cooperation (mechanism design) and multi-agent systems.

Boaz Ben-Moshe Boaz Ben-Moshe is a faculty member in the Department of Computer in Ariel University Center, Israel. He received the B.Sc., M.Sc., and Ph.D. degrees in Computer Science from Ben-Gurion University, Israel. During 2004-2005 he was a post-doctoral fellow at Simon Fraser University, Vancouver, Canada. His main research areas are: Computational Geometry and GIS algorithms. His research includes Geometric data compression, Optimization of wireless networks, Computing visibility graphs, and Vehicle routing problems. In 2008 He has founded the Kinematics and Computational Geometry Laboratory with Dr. Nir Shvalb, see: `http://www.ariel.ac.il/sites/kcg`.

Correspondence and offprint requests to: Eran Shaham, Department of Computer Science, Bar-Ilan University, Ramat-Gan, 52900 Israel. Email: erans@macs.biu.ac.il