Dominant Set Biclustering

Matteo Denitto[1(✉)], Manuele Bicego[1], Alessandro Farinelli[1],
and Marcello Pelillo[2]

[1] Department of Computer Science, University of Verona, Verona, Italy
matteo.denitto@univr.it
[2] ECLT - University of Venice, Venice, Italy

Abstract. Biclustering, which can be defined as the simultaneous clustering of rows and columns in a data matrix, has received increasing attention in recent years, being applied in many scientific scenarios (e.g. bioinformatics, text analysis, computer vision). This paper proposes a novel biclustering approach, which extends the *dominant-set clustering* algorithm to the biclustering case. In particular, we propose a new way of representing the problem, encoded as a graph, which allows to exploit dominant set to analyse both rows and columns simultaneously. The proposed approach has been tested by using a well known synthetic microarray benchmark, with encouraging results.

1 Introduction

Biclustering, also widely known as co-clustering, can be defined as the simultaneous clustering of both rows and columns of a given data matrix [5,12,17]. With respect to clustering, the main differences of biclustering consist in the exploitation of *local information* (instead of global) *to retrieve subsets of rows sharing a "similar" behaviour in a subsets of columns, and vice versa* (instead of subsets of rows sharing a similar behaviour among *whole* the columns). Although bi-clustering was born and mainly applied to analyse gene expression microarray data [5,22], it has been recently exploited in a more various range of applications from clickstream data [18], passing by recommender systems [19], to different Computer Vision scenarios (such as facial expression recognition [16], motion and plane estimation [8]).

Different biclustering techniques have been proposed in the past – for a comprehensive review please refer to [12,17,22,23] – each one characterized by different features, such as computational complexity, effectiveness, interpretability and optimization criterion. Various of such previous approaches are based on the idea of adapting a given clustering technique to the biclustering problem, for example by repeatedly performing rows and columns clustering [10,14].

This paper follows the above-described research trend, and proposes a novel biclustering algorithm, which extends and adapts to the biclustering scenario the *dominant-set* based clustering. The concept of dominant set can be depicted from various points of view, since it involves optimization theory, graph theory,

M. Pelillo and E. Hancock (Eds.): EMMCVPR 2017, LNCS 10746, pp. 49–61, 2018.
https://doi.org/10.1007/978-3-319-78199-0_4

game theory and pattern recognition. Approaching it from a clustering perspective, given a set of objects V to group, a dominant set $C \subseteq V$ is a subset of objects with two well-defined properties: (i) all elements belonging to C should be highly similar to each other, and (ii) no larger cluster should contain C as a proper subset. Thus, C can be informally expressed as a *maximally coherent* set of data items [4,21]. Practically, a dominant set C is represented by a characteristic vector $\mathbf{x}$ where an entry x_i represents how likely the object v_i belongs to the retrieved cluster. In [4,21] authors provide a clustering algorithm based on dominant sets which is theoretically solid and supported by several experimental evaluations. Differently from classical clustering approaches, dominant-set clustering does not provide a partition of the data, and it can be successfully exploited in highly noisy contexts or scenarios with outliers. Moreover, dominant-set clustering can be exploited also in cases where the similarity matrix between objects is asymmetric. These two last considerations provide a solid link between dominant-set and classical biclustering approaches which, in most of the cases, do not provide a partition of the data matrix, and deal with non-squared matrices (hence, not symmetric) [17].

A first step toward the usage of dominant sets in the biclustering scenario has been presented in [26]. In this case authors propose to retrieve the bicluster by iteratively sorting and shifting the rows/columns of the given data matrix. Such sorting and shifting is made according to the dominant-set characteristic vector. However this algorithm do not exploit the potential of dominant set to group both rows and columns simultaneously, which is the core of classic biclustering algorithms and of this paper. Another approach similar in spirit is the one presented in [9]. In fact, authors of [9] tackle biclustering by retrieving *bi-cliques* on a bipartite-graph adjacency matrix. Although the technique derived shows some similarities with the one proposed in this paper, the resulting technique does not provide dominant sets as results and for this reason we do not present its details in this manuscript.

For this reason we decided to investigate how the concept of dominant set can be extended to the biclustering scenario. Among the different biclustering algorithm typologies, one branch involves the representation of the problem as a edge-cutting problem in a weighted bipartite graph, where one set of nodes represents rows and the other represents the columns [1,13]. However, a dominant set from a graph theory perspective is equivalent to *maximal clique* [4], and thus it cannot be applied directly on a bipartite graph (since a maximal clique in a bipartite graph is composed by only two nodes). We thus provide a novel simple graph representation for the biclustering problem. This involves the exploitation of data matrix entries as a *similarity* measure between rows-columns couples. The intuition behind this usage of the data matrix comes from the consideration that in various biclustering scenarios (such as recommender systems, gene expression analysis, clickstream data) the information encoded in the data matrix represents how much a row "is important" for that particular column. Moreover, to obtain a theoretically appropriate dominant set, we modified the bipartite graph adjacency matrix following the baselines provided in [21,29].

We evaluate the performance of the proposed approach on both synthetic and real datasets, favourably comparing with the current the state-of-the-art.

The remainder of paper is organized as follows: Sect. 2 introduces the dominant-set clustering approach, clearing the connections with other fields; our algorithm is then described in Sect. 3, whereas the experimental evaluation is given in Sect. 4; finally Sect. 5 concludes the paper.

2 Dominant Set Clustering

In this section we summarize the contributions provided in [4], providing the background knowledge concerning the dominant-set clustering algorithm.

Clustering is the problem of organizing a set of data elements into groups in a way that each group satisfies an *internal homogeneity* and *external inhomogeneity* property. Differently from classical clustering approaches the dominant-set algorithm faces the problem from a game theory perspective, instantiating a *non-cooperative clustering game* where the notion of a cluster turns out to be equivalent to a classical equilibrium concept from (evolutionary) game theory, as the latter reflects both the internal and external cluster conditions [4]. As discussed in Sect. 1, the internal condition asserts that elements belonging to the cluster should have high mutual similarities, whereas the external property claims that a cluster cannot be further extended by introducing external elements.

Formally, let $G = (V, E, \omega)$ be a weighted graph representing a clustering problem instance, where $V = \{1, \ldots, n\}$ is a finite set of vertices (representing the objects to group), $E \subseteq V \times V$ and $\omega : E \to \mathbb{R}$. We adopt $A_{ij} = \omega(i, j)$ to denote the graph adjacency matrix, representing the objects similarities. Given a non-empty subset of objects $C \subseteq V$, we define the *average weighted in-degree* of $i \in V$ with respect to C as:

$$\text{awindeg}_C(i) = \frac{1}{|C|} \sum_{j \in C} A_{ij},$$

where $|C|$ is the cardinality of C. Also, if $j \in C$ we define

$$\phi_C(i, j) = A_{ij} - \text{awindeg}_C(j),$$

which is a measure of the relative similarity of object i with object j with respect to the average similarity of object j with elements in C. Let us define the weight of an object i with respect to a set C:

$$W_C(i) = \begin{cases} 1, & \text{if } |C| = 1 \\ \sum_{j \in C \setminus \{i\}} \phi_{C \setminus \{i\}}(i, j) W_{C \setminus \{i\}}(j), & \text{otherwise.} \end{cases}$$

Please note that such definition is inductive and not circular, since in the sum j takes belongs to C excluding the interested point i. Now, we define the total weight of C as:

$$W(C) = \sum_{i \in C} W_C(i).$$

The weight $W_C(j)$ captures the strength of the coupling between vertex j and the other elements in the set relative to the overall coupling among the vertices. Such properties of the weighting are the basis for the formalizations of the notion of dominant set as a notion of a cluster.

Definition 1. *A non-empty subset of objects $C \subseteq V$ such that $W(T) > 0$ for any non-empty $T \subseteq C$, is said to be a* dominant set *if*

1. $W_C > 0$*, for all* $i \in C$*,*
2. $W_{C \cup i}(i) < 0$*, for all* $i \notin C$*.*

This definition provides conditions that correspond to the two main properties of a cluster: the internal and external conditions mentioned above.

Such formulation of clustering has connections with other scientific fields (namely optimization theory, graph theory and game theory), this links have been theoretically proven in [4]. The understanding of these links are needed to perceive how authors in [4] decide to compute a dominant set/cluster from the similarity matrix A. Moreover, the basis of how we decided to extend such approach to biclustering are theoretically founded in the theorems lying behind such connections.

Summarizing, in [4] they show that:

– with respect to *optimization theory*: a dominant set can be can be characterized in terms of local solutions pf the following standard quadratic program

$$\begin{array}{ll} \text{maximize} & f(\mathbf{x}) = \mathbf{x}^T A\mathbf{x} \\ \text{subject to} & \mathbf{x} \in \Delta \subset \mathbb{R}^n, \end{array} \tag{1}$$

where

$$\Delta = \left\{ \mathbf{x} \in \mathbb{R}^n : \sum_{j \in V} x_j = 1 \text{ and } x_j \geq 0 \text{ for all } j \in V \right\}.$$

And, particularly, they show that if C is a dominant set of A, then its characteristic vector $\mathbf{x}^C$ is a strict local solution to (1). Conversely, if $\mathbf{x}^*$ is a strict local solution to (1) then its support $\sigma = \sigma(\mathbf{x}^*)$ is a dominant set of A. Where $\sigma(\mathbf{x})$ is defined as the index set of the positive components in $\mathbf{x}$.

– with respect to *graph theory*: a dominant set of A corresponds to a *maximal clique* in the corresponding graph. This means that the nodes corresponding to the support vector of $\mathbf{x}^C$ are a maximal clique, and thus they are all connected to each other and such clique cannot be expanded.

– with respect to *game theory*: if C a dominant set of A, then its characteristic vector $\mathbf{x}^C$ is an *Evolutionary Stable Strategy* of the corresponding *clustering game*[1].

[1] The idea is to setup a symmetric, non-cooperative game, called clustering game, between two players. Data points V are the strategies available to the players and the similarity matrix A encodes their payoff matrix.

Please for all the details and demonstration concerning these connections refer to [4].

Finally, once the problem has been theoretically instantiated, the authors of [4] provide two different approaches to retrieve a dominant set from A. Both strategies have roots in the game theory domain and it should not be surprising that methods developed in this context can be used to find dominant sets, given the tight relation that exists between dominant sets and the game-theoretic notion of equilibrium. The first involves *replicator dynamics* [24], whereas the other concerns *infection and immunization dynamics* [3].

3 Biclustering with Dominant Set

In this Section we provide our formulation of the biclustering problem, including the details of how dominant set are extended to such scenario.

As mentioned in Sect. 1, biclustering aims at the simultaneous clustering of rows and columns of a given data matrix. Formally, we denote as $D \in \mathbb{R}^{n \times m}$ the given data matrix, and let $R = \{1, \dots, n\}$ and $K = \{1, \dots, m\}$ be the set of row and column indices. We adopt D_{TL}, where $T \subseteq R$ and $L \subseteq K$, to represent the submatrix with the subset of rows in T and the subset of columns in L. Given this notation, we can define a *bicluster* as a submatrix D_{TL}, such that the subset of rows of D with indices in T exhibits a "coherent behavior" (in some sense) across the set of columns with indices in L, and vice versa. The choice of coherence criterion defines the type of biclusters to be retrieved (for a comprehensive survey of biclustering criteria, see [17,20]).

In this paper we propose to tackle biclustering exploiting the principles of dominant set definition. Although a preliminary approach toward this direction has already be presented in literature, authors of [26] present an iterative rows/columns clustering algorithm which does not fully exploit dominant set potentials. Specifically, the technique proposed in [26] defines a weighted correlation measure adopted to build a similarity matrix between the rows of the given data matrix. On such matrix authors apply dominant-set clustering and they exploit the characteristic vector $\mathbf{x}^C$ to sort the data matrix rows. This result in a data matrix where rows belonging to the bicluster are shifted to the bottom. At this point they compute a similarity matrix for the columns adopting $\mathbf{x}^C$ as weight for the correlation, giving more importance to the rows belonging to the biclusters. At this point dominant-set clustering is applied on the columns similarity matrix. The idea is that weighting the columns correlation with respect to the characteristic vector (computed on the rows) should help in retrieving a subset of columns acting similarly in that particular subset of rows. Columns are then shifted according to their characteristic vector and such operations is iteratively repeated twice for the rows and twice for the columns [26]. The resulting data matrix now contains the bicluster in the bottom-right position. To retrieve the actual bicluster authors compute the correlation between consecutive rows (starting from the bottom), and they stop when such correlation is below a certain threshold (same procedure applies for retrieving bicluster columns).

Hence, in [26], authors exploit the result of dominant-set clustering to iteratively order rows and columns to obtain a data matrix where the bicluster is isolated in the bottom-right portion of the matrix.

One branch of techniques exploits the weighted bipartite graph representation to face biclustering. In this context the common choice is to represent with two distinct sets of nodes the rows R and the columns K of the data matrix. Then, connect with edges only nodes belonging to different sets, and the weights are none other than the data matrix entries. Given this graph, the problem is thus formulated as an edge cutting problem where the surviving edges define the rows and the columns belonging to the bicluster [1]. Such cutting is obviously guided by a pre-defined objective function.

Alternatively to what previously presented, what we propose in this paper is to adopt a graph where rows and columns are represented by a unique set of nodes. Hence, given a data matrix D, we instantiate the biclustering problem as a graph $G = (V, E, \omega)$ where the vertices $V = \{1, \ldots, n + m\}$ represent the rows ($\{v_1, \ldots, v_n\}$) and columns ($\{v_{n+1}, \ldots, v_{n+m}\}$) of D. With this representation, we can easily encode the bipartite graph mentioned above. In fact, this can be obtained by introducing the data matrix D in correspondence of the positions connecting rows and columns in the adjacency matrix A. Practically, $A([1, \ldots, n], [n+1, \ldots, n+m]) = D$ and, to obtain a consistent adjacency matrix, we also define $A([n + 1, \ldots, n + m], [1, \ldots, n]) = D^T$. The other portions of A, representing row-row and column-column similarities, are set to 0 (resulting in no edges connecting such vertices).

Once the bipartite graph is represented through a squared similarity matrix, we can now exploit dominant set definitions (usually applied in clustering context) to obtain a bicluster. In fact, since the built adjacency matrix contains high values only in row-columns positions (assuming a positive data matrix D), we expect the dominant set to be a group of rows presenting high similarities in a subset of columns (and vice versa). However, recalling the link between dominant set and graph theory (presented in Sect. 2), a dominant set of A is equivalent to a maximal clique in the correspondent graph. Hence, since the graph is bipartite (no row-row/column-column edges), a maximal clique is composed by just two nodes: one row and one column. Particularly, by adding another node we cannot have a clique since the subset of nodes will lack of one edge.

To overcome this, we add a negative value $-\alpha$ (where $\alpha \geq 0$) on the main diagonal of the similarity matrix A. This has been proved to be equivalent to solve a standard quadratic problem where the values of the off-diagonal entries of A are increased by α, and the main diagonal is set to 0 [21,29]. Note that adding α on the off-diagonal entries introduces edges between row-row and column-column nodes, resulting in a classic graph (not bipartite). Thus applying dominant set on this latter version of A results in an actual maximal clique, where a subset of rows will be selected simultaneously with a subset of columns. We depicted how the similarity matrix A is built in Fig. 1.

Intuitively, this is obtained because, independently by value adopted for α, the actual information is still contained in the rows-columns portions of A

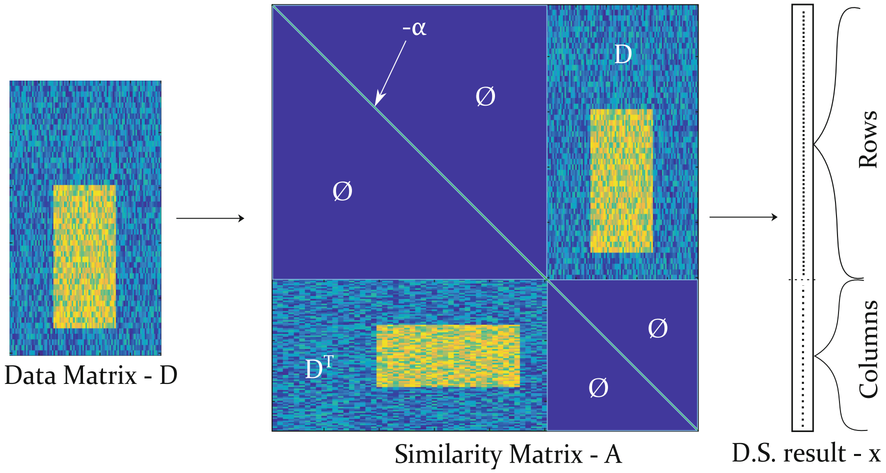


Fig. 1. Scheme of the algorithm

(in fact the value of α is the same for all the entries, and hence it is not informative). It has also been theoretically proved that increasing the value of α will increase the dimension of the resulting clique [4,21,29], as shown in Fig. 2.

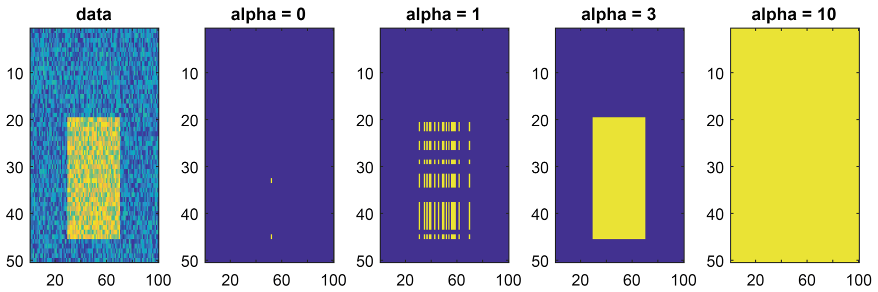


Fig. 2. Different results varying alphas

Summarizing, given a data matrix D with n rows and m columns we represent the biclustering problem as a graph having $n + m$ vertices, where the first n represent the rows and the remaining m represent the columns. Rows and columns are connected to each other with weights corresponding to the entries of D, whereas row-row and column-column edges have weights equal to α. A dominant set in such graph is a maximal clique isolating group of rows presenting high similarities in a group of columns, hence a bicluster.

To obtain such bicluster/dominant set we resort to *replicator dynamics*. The replicator dynamics are deterministic game dynamics that have been developed in evolutionary game theory. It considers an idealized scenario whereby individuals are repeatedly drawn at random from a large, ideally infinite, population to play a two-player game. In contrast to classical game theory, here players are not supposed to behave rationally or to have complete knowledge of the details of the game. They act instead according to an inherited behavioral pattern, or pure strategy, and it is supposed that some evolutionary selection process operates over time on the distribution of behaviors. Particularly, we adopt the iterative discrete-time replicator dynamics, which are given by

$$x_i(t+1) = x_i(t)\frac{(A\mathbf{x}(t))_i}{\mathbf{x}(t)^T A\mathbf{x}(t)}, \tag{2}$$

for $i \in V$. For further details concerning the theoretical basis relying under the connections between replicator dynamics and dominant set, we refer interested readers to the recent summary [4].

The resulting algorithm – called *Dominant Set Biclustering (DSB)* – is thus parsimonious in terms of space $\mathcal{O}(n+m)$ and efficient in terms of time of execution $\mathcal{O}(n+m)$. The parameters of such approach are: (i) the off-diagonal value α, (ii) the convergence of the replicator dynamics (which can be defined with a maximum number of iterations or with a threshold between consecutive changing of $\mathbf{x}$.

Please note that, although the method recovers one bicluster at a time (as widely exploited in literature [2,6,7]), there exists different heuristic to “mask” the obtained bicluster and to look for the next one. Specifically, to mask the retrieved bicluster we put zeros in the corresponding positions inside the adjacency matrix A.

4 Experimental Evaluation

The proposed approach has been evaluated using two sets of synthetic datasets and one Computer Vision dataset divided in two problems.

4.1 Synthetic Experiments

The two synthetic benchmarks are created to simulate gene expression matrices containing a single bicluster. In the first dataset the implanted biclusters are constant valued bicluster (we call this “Constant Bicluster Benchmark”), while in the second dataset additively coherent biclusters were used (we call this “Evolutionary Bicluster Benchmark”).

In both cases, each matrix has been generated using the following procedure: (i) we generate a 50×50 matrix containing random values uniformly distributed between 0 and 1; (ii) we insert a constant valued (or additively coherent valued) bicluster, whose dimension was 25% of the matrix size, the bicluster was inserted in a random position; (iii) finally, the entire matrix has been perturbed with Gaussian noise. The standard deviation of the Gaussian noise is a percentage of the difference between the mean of the entries belonging to the bicluster and the mean of the background. 5 different noise levels (i.e. percentages) were used, ranging from 0 (no noise) to 0.2 (high noise). For each noise level, 30 matrices have been generated, resulting in a total of 75 matrices.

The quality of the retrieved biclusters have been assessed using two standard indices, also employed in [28]: (i) *purity*: percentage of points retrieved by the algorithms which actually belong to the real bicluster; (ii) *inverse purity*: percentage of points belonging to the true bicluster which have been retrieved by the algorithms. Calling C the bicluster found by the algorithm and L the ground truth, the indices are calculated as follows:

$$\text{Purity} = \frac{|C \cap L|}{|C|}, \qquad \text{Inverse Purity} = \frac{|L \cap C|}{|L|}.$$

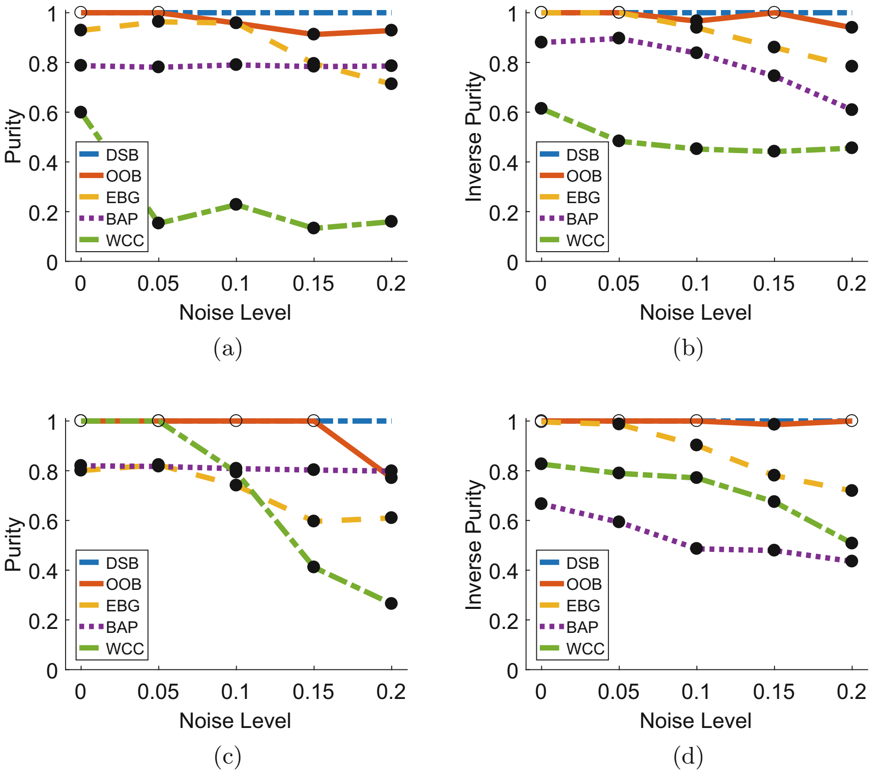
The proposed approach has been compared with four other biclustering algorithms, including the preliminary one adopting dominant set (mentioned in Sect. 1). The results for the OOB, EBG and BAP algorithms have been taken from [8], whereas for WCC (the first approach resorting to dominant set) we implemented the code following the indications presented in [26] and adopting the suggested values for the parameters.

The results for the Constant and Evolutionary Bicluster benchmarks are shown in Fig. 3, where purity (a, c) and inverse purity (b, d) are displayed for the different methods, while varying the noise level. Each point represents the average over the 30 runs of the given noise level. In the plot, a full marker indicates that the difference between the considered method and the proposed approach is statistically significant[2].

The results evidently show that the proposed approach significantly outperforms the current state-of-the-art, especially when the level of noise increases, thus confirming the potentials of dominant sets in complex highly noisy situations. Concerning WCC, it is expectable that the performances on the constant bicluster benchmark are influenced by the exploitation of the weighted correlation coefficient. In fact, supposing to correctly select the columns involved in the bicluster, the behaviour of the bicluster in the selected columns and the one of the background is similar (since the value of the bicluster is constant). Thus, it is difficult for the method to differentiate between these two situations. This is also confirmed by the better performance of WCC in the evolutionary bicluster benchmark, where background and bicluster have different behaviours (background is constant and the bicluster evolves). However, in both cases the proposed approach provides better quality results, demonstrating that the exploitation of a more solid framework involving dominant set is sound.

Multiple Structure Recovery Dataset. *Multiple structure recovery* (MSR) concerns the extraction of multiple models from noisy or outlier-contaminated data. MSR is an important and challenging problem, which emerges in many computer vision applications [11,15,27]. In general, an instance of an MSR

[2] We performed a t-test for each noise level (on the result of the 30 matrices), we set the significance level to 5%.



(a) (b)

(c) (d)

Fig. 3. Purity (a, c) and Inverse Purity (b, d) for matrices with constant (a, b) and additive coherent (c, d) biclusters

problem is represented by a *preference matrix* containing, in one dimension, the points under analysis, and in the other, the hypotheses/structures to which points should belong. The entry (i, j) in this matrix indicates how well a certain point i is represented by the given hypothesis/structure j.

The Adelaide dataset, which has been already exploited for assessing quality of biclustering algorithms [8], involves two type of MSR problems: motion and plane estimation. Given two different images of the same scene, where several objects move independently, motion segmentation aims at recovering subsets of point matches that undergo the same motion. Given two uncalibrated views of a scene, plane segmentation consists in retrieving the multi-planar structures by fitting homographies to point correspondences. The AdelaideRMF dataset[3] is composed of 38 image pairs (19 for motion segmentation and 19 for plane segmentation), with matching points contaminated by strong outliers. The ground-truth segmentations are also available. As in [8,25], we adopt the misclassification errors to assess the results.

[3] https://cs.adelaide.edu.au/~hwong/doku.php?id=data.

Table 1 presents the results. We report two different results for DSB; we run the algorithm varying the parameters, and on the basis of the considered results the performances can slightly vary. The last columns of Table 1 (*DSB best*) shows the results for DSB where we consider for each different matrix the best performance with respect to the misclassification error (varying the parameters). The results in the sixth column (*DSB best set*), which are slightly worse than the previous, are obtained by selecting the best set of parameters values minimizing the misclassification error (one for the motion segmentation and one for the plane estimation). Please note that two columns slightly differs, demonstrating that dominant sets strongly resist to both noise and the massive presence of outliers. With respect to other techniques, Table 1 shows that the proposed approach improves the results of the state-of-the-art in the plane segmentation dataset, and that we also provide comparable result on the motion segmentation dataset.

Table 1. Misclassification error (ME %) for motion segmentation (above) and planar segmentation (bottom). k is the number of models and % out is the percentage of outliers.

	k	%out	T-lnkg	RCMSA	RPA	DSB best set	DSB best
biscuitbookbox	3	37.21	3.10	16.92	3.88	10.42	6.17
breadcartoychips	4	35.20	14.29	25.69	7.50	5.48	5.48
breadcubechips	3	35.22	3.48	8.12	5.07	5.21	5.21
breadtoycar	3	34.15	9.15	18.29	7.52	11.44	11.44
carchipscube	3	36.59	4.27	18.90	6.50	4.24	4.24
cubebreadtoychips	4	28.03	9.24	13.27	4.99	9.48	9.48
dinobooks	3	44.54	20.94	23.50	15.14	14.16	14.16
toycubecar	3	36.36	15.66	13.81	9.43	16.00	16.00
biscuit	1	57.68	16.93	14.00	1.15	16.36	16.36
biscuitbook	2	47.51	3.23	8.41	3.23	2.63	2.63
boardgame	1	42.48	21.43	19.80	11.65	8.96	8.96
book	1	44.32	3.24	4.32	2.88	10.69	10.69
breadcube	2	32.19	19.31	9.87	4.58	11.57	9.50
breadtoy	2	37.41	5.40	3.96	2.76	3.12	3.12
cube	1	69.49	7.80	8.14	3.28	3.31	3.31
cubetoy	2	41.42	3.77	5.86	4.04	4.81	4.81
game	1	73.48	1.30	5.07	3.62	1.71	1.71
gamebiscuit	2	51.54	9.26	9.37	2.57	4.57	4.57
cubechips	2	51.62	6.14	7.70	4.57	7.04	7.04
mean			9.36	12.37	5.49	7.96	7.62
median			7.80	9.87	4.57	7.04	6.17

	k	%out	T-lnkg	RCMSA	RPA	DSB best set	DSB best
unionhouse	5	18.78	48.99	2.64	10.87	25.00	25.00
bonython	1	75.13	11.92	17.79	15.89	4.04	4.04
physics	1	46.60	29.13	48.87	0.00	2.83	0.94
elderhalla	2	60.75	10.75	29.28	0.93	5.14	2.80
ladysymon	2	33.48	24.67	39.50	24.67	10.54	10.54
library	2	56.13	24.53	40.72	31.29	13.95	13.95
nese	2	30.29	7.05	46.34	0.83	0	0
sene	2	44.49	7.63	20.20	0.42	0.40	0
napiera	2	64.73	28.08	31.16	9.25	13.24	13.24
hartley	2	62.22	21.90	37.78	17.78	3.12	1.56
oldclassicswing	2	32.23	20.66	21.30	25.25	8.44	8.44
barrsmith	2	69.79	49.79	20.14	36.31	51.03	51.03
neem	3	37.83	25.65	41.45	19.86	25.72	15.76
elderhallb	3	49.80	31.02	35.78	17.82	25.88	18.82
napierb	3	37.13	13.50	29.40	31.22	20.84	20.84
johnsona	4	21.25	34.28	36.73	10.76	20.37	20.37
johnsonb	7	12.02	24.04	16.46	26.76	19.87	19.87
unihouse	5	18.78	33.13	2.56	5.21	3.69	3.69
bonhall	6	6.43	21.84	19.69	41.67	38.76	38.76
mean			24.66	28.30	17.20	15.41	14.19
median			23.38	29.40	17.53	13.24	13.24

5 Conclusions

In this paper we proposed a novel algorithm facing the biclustering problem. Such algorithm extends the definition of dominant sets (already exploited for clustering) to the biclustering scenario. It involves a novel paradigm to represent the problem, coupled with solid theoretical basis. Dominant sets representing the bicluster are efficiently computed resorting to discrete-time replicator dynamics. The algorithm performances have been assessed on both synthetic and real datasets, providing better quality solutions when compared with the state-of-the-art.

References

1. Ahmad, W., Khokhar, A.: cHawk: an efficient biclustering algorithm based on bipartite graph crossing minimization. In: VLDB Workshop on Data Mining in Bioinformatics (2007)
2. Ben-Dor, A., Chor, B., Karp, R., Yakhini, Z.: Discovering local structure in gene expression data: the order-preserving submatrix problem. J. Comput. Biol. **10**(3–4), 373–384 (2003)
3. Bulò, S.R., Bomze, I.M.: Infection and immunization: a new class of evolutionary game dynamics. Games Econ. Behav. **71**(1), 193–211 (2011)
4. Bulò, S.R., Pelillo, M.: Dominant-set clustering: a review. Eur. J. Oper. Res. **262**(1), 1–13 (2017)
5. Cheng, Y., Church, G.: Biclustering of expression data. In: Proceeding of Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB00), pp. 93–103 (2000)
6. Cheng, Y., Church, G.M.: Biclustering of expression data. In: ISMB, vol. 8, pp. 93–103 (2000)
7. Denitto, M., Farinelli, A., Figueiredo, M.A., Bicego, M.: A biclustering approach based on factor graphs and the max-sum algorithm. Pattern Recogn. **62**, 114–124 (2017)
8. Denitto, M., Magri, L., Farinelli, A., Fusiello, A., Bicego, M.: Multiple structure recovery via probabilistic biclustering. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) S+SSPR 2016. LNCS, vol. 10029, pp. 274–284. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49055-7_25
9. Ding, C., Zhang, Y., Li, T., Holbrook, S.R.: Biclustering protein complex interactions with a biclique finding algorithm. In: 2006 Sixth International Conference on Data Mining, ICDM 2006, pp. 178–187. IEEE (2006)
10. Farinelli, A., Denitto, M., Bicego, M.: Biclustering of expression microarray data using affinity propagation. In: Loog, M., Wessels, L., Reinders, M., Ridder, D. (eds.) Pattern Recognition in Bioinformatics. Lecture Notes in Computer Science, vol. 7036, pp. 13–24. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24855-9_2
11. Fitzgibbon, A.W., Zisserman, A.: Multibody structure and motion: 3-D reconstruction of independently moving objects. In: Vernon, D. (ed.) ECCV 2000. LNCS, vol. 1842, pp. 891–906. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45054-8_58
12. Flores, J.L., Inza, I., Larraaga, P., Calvo, B.: A new measure for gene expression biclustering based on non-parametric correlation. Comput. Methods Programs Biomed. **112**(3), 367–397 (2013). http://www.sciencedirect.com/science/article/pii/S0169260713002605
13. Gao, B., Liu, T.Y., Zheng, X., Cheng, Q.S., Ma, W.Y.: Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pp. 41–50. ACM (2005)
14. Getz, G., Levine, E., Domany, E.: Coupled two-way clustering analysis of gene microarray data. Proc. Nat. Acad. Sci. USA **97**(22), 12079–12084 (2000)
15. Häne, C., Zach, C., Zeisl, B., Pollefeys, M.: A patch prior for dense 3D reconstruction in man-made environments. In: 2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), pp. 563–570. IEEE (2012)

16. Khan, S., Chen, L., Zhe, X., Yan, H.: Feature selection based on co-clustering for effective facial expression recognition. In: 2016 International Conference on Machine Learning and Cybernetics (ICMLC), pp. 48–53. IEEE (2016)
17. Madeira, S., Oliveira, A.: Biclustering algorithms for biological data analysis: a survey. IEEE Trans. Comput. Biol. Bioinf. **1**, 24–44 (2004)
18. Melnykov, V.: Model-based biclustering of clickstream data. Comput. Stat. Data Anal. **93**, 31–45 (2016)
19. Mukhopadhyay, A., Maulik, U., Bandyopadhyay, S., Coello, C.A.C.: Survey of multiobjective evolutionary algorithms for data mining: part ii. IEEE Trans. Evol. Comput. **18**(1), 20–35 (2014)
20. Oghabian, A., Kilpinen, S., Hautaniemi, S., Czeizler, E.: Biclustering methods: biological relevance and application in gene expression analysis. PloS One **9**(3), e90801 (2014)
21. Pavan, M., Pelillo, M.: Dominant sets and hierarchical clustering, p. 362. IEEE (2003)
22. Pontes, B., Giráldez, R., Aguilar-Ruiz, J.S.: Biclustering on expression data: a review. J. Biomed. Inf. **57**, 163–180 (2015)
23. Prelic, A., Bleuler, S., Zimmermann, P., Wille, A., Bhlmann, P., Gruissem, W., Hennig, L., Thiele, L., Zitzler, E.: Comparison of biclustering methods: a systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics **22**(9), 1122–1129 (2006)
24. Smith, J.M.: Evolution and the theory of games. In: Smith, J.M. (ed.) Did Darwin Get It Right?, pp. 202–215. Springer, Boston (1988). https://doi.org/10.1007/978-1-4684-7862-4_22
25. Soltanolkotabi, M., Elhamifar, E., Candès, E.J.: Robust subspace clustering. Ann. Stat. **42**(2), 669–699 (2014)
26. Teng, L., Chan, L.: Discovering biclusters by iteratively sorting with weighted correlation coefficient in gene expression data. J. Sig. Process. Syst. **50**(3), 267–280 (2008)
27. Toldo, R., Fusiello, A.: Image-consistent patches from unstructured points with J-linkage. Image Vis. Comput. **31**(10), 756–770 (2013)
28. Tu, K., Ouyang, X., Han, D., Honavar, V.: Exemplar-based robust coherent biclustering. In: SDM, pp. 884–895. SIAM (2011)
29. Zemene, E., Pelillo, M.: Interactive image segmentation using constrained dominant sets. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 278–294. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_17