Int. J. Appl. Math. Comput. Sci., 2018, Vol. 28, No. 4, 771–786
DOI: 10.2478/amcs-2018-0059

CLUSTERING BASED ON EIGENVECTORS OF THE ADJACENCY MATRIX

MAŁGORZATA LUCIŃSKA [a,*], SŁAWOMIR T. WIERZCHOŃ [b]

[a] Department of Management and Computer Modelling
Kielce University of Technology, Al. 1000-lecia PP 7, 25-314 Kielce, Poland
e-mail: lucinska@tu.kielce.pl

[b] Institute of Computer Science
Polish Academy of Sciences, ul. Jana Kazimierza 5, 01-248 Warsaw, Poland

The paper presents a novel spectral algorithm EVSA (eigenvector structure analysis), which uses eigenvalues and eigenvectors of the adjacency matrix in order to discover clusters. Based on matrix perturbation theory and properties of graph spectra we show that the adjacency matrix can be more suitable for partitioning than other Laplacian matrices. The main problem concerning the use of the adjacency matrix is the selection of the appropriate eigenvectors. We thus propose an approach based on analysis of the adjacency matrix spectrum and eigenvector pairwise correlations. Formulated rules and heuristics allow choosing the right eigenvectors representing clusters, i.e., automatically establishing the number of groups. The algorithm requires only one parameter—the number of nearest neighbors. Unlike many other spectral methods, our solution does not need an additional clustering algorithm for final partitioning. We evaluate the proposed approach using real-world datasets of different sizes. Its performance is competitive to other both standard and new solutions, which require the number of clusters to be given as an input parameter.

Keywords: spectral clustering, adjacency matrix eigenvalues/eigenvectors, graph perturbation theory, eigengap heuristics.

1. Introduction

Cluster analysis is concerned with the division of data into homogeneous groups (clusters) such that data items within one cluster are similar to each other, and those within different clusters are dissimilar. All clustering algorithms aim at one or both of the following goals: to find a proper partition and to determine the number of clusters.

Spectral clustering techniques belong to the most popular and efficient clustering methods (see, e.g., von Luxburg, 2007; Jia *et al.*, 2014). In these methods compactness (characteristic of K-means algorithms) is replaced by connectivity, which better identifies clusters in many cases. Spectral algorithms arise from concepts developed within spectral graph theory (Chung, 1997). The basic idea is to represent relationships between the elements of the dataset by a similarity matrix which can be viewed as a generalized adjacency matrix, defining the so-called similarity graph. The nodes in this graph represent elements of the dataset and each edge reflects the similarity between the pair of appropriate elements. Selected eigenvectors of the affinity (or some its transformation) matrix provide a new coordinate system of reduced dimensionality. It is important to note that spectral clustering can be applied to the case of metric and nonmetric data. In fact, representation of data is irrelevant for spectral clustering: what we operate with is the similarity matrix. The elements of this matrix can be computed automatically (e.g., as a transformation of a distance between elements with metric representation) or attached subjectively in the case of nonmetric data.

There are two main approaches used in spectral clustering. The first one relies upon recursively splitting a given group into two smaller groups until the desired partition is obtained. Here a single eigenvector gives us guidance on how to partition the group, and initially the whole dataset is considered such a group. The other solution involves taking a few eigenvectors and applying a simplified clustering algorithm, such as K-means, to divide the data described by the coordinates derived from these eigenvectors. Both methods require prior knowledge

*Corresponding author

of the number of clusters K or a quality function that indicates the end of a recursive partitioning procedure. However, some recent spectral solutions try to detect the number of clusters automatically. The algorithm proposed by Azran and Ghahramani (2006) makes use of the relation between the magnitude of the eigenvalues and the Gaussian scaling parameter in order to address the issue. Zheng and Lin (2004) estimated cluster numbers also with the help of the scaling parameter σ from the Gaussian kernel

$$K_G(x,y) = \exp\Big(-\frac{\|x-y\|^2}{2\sigma^2}\Big). \tag{1}$$

They established the number of clusters on the basis of the eigengap, i.e., the difference between two successive eigenvalues, sorted in ascending order. The right number of clusters should remain the same for relatively big changes in the Gaussian scaling parameter σ.

Zelnik-Manor and Perona (2004) proposed an iterative method based on eigenvectors. For each integer k, it creates an eigenvector matrix $\mathbf{X}$ and tries to find a rotation $\mathbf{R}$ such that each row in the matrix $\mathbf{XR}$ has a single nonzero entry. It is possible to fulfill the condition only when the number of eigenvectors equals that of clusters. Taking more eigenvectors will result in more than one nonzero entry in some of the rows. The smaller number of eigenvectors fails to create a full basis spanning the subspace of the canonical coordinate system.

Another problem with spectral clustering methods concerns the choice of the right eigenvectors, which, enable proper partitions. Numerical experiments suggests that it is not guaranteed that the largest K eigenvectors can well detect the structure of the data. Xiang and Gong (2008) were the first to use eigenvector selection to improve the performance of the spectral clustering method. Liu *et al*. (2014) proposed an eigenvector selection method based on latent tree models. For each integer k from an established range, they built a latent tree model using k eigenvectors. In order to determine the number of clusters, the BIC score (Schwarz, 1978) was used as the performance index.

Shi *et al*. (2009) developed theoretical results to use clustering information contained in the eigenvectors of the adjacency matrix based on a radial kernel function with a sufficiently fast tail decay. They designed a data spectroscopic clustering (`DaSpec`) algorithm that utilizes properly selected eigenvectors to determine the number of clusters automatically and to group the data accordingly.

In this paper we present a solution which addresses all the above mentioned challenges. It is similar to the `DaSpec` algorithm. It does not require a user to know the number of clusters *a priori*, selects the right eigenvectors and does not need any additional clustering algorithms nor partitioning quality functions. The solution is based on some geometric properties of the eigenvectors of an adjacency matrix. Careful analysis of the eigenvectors allows one to obtain proper partitioning without the need to give the number of clusters or to use additional tools for clustering.

We also increased the flexibility of the `DaSpec` algorithm by taking into consideration a larger group of eigenvectors and distinguishing between two subsets on the basis of eigenvector entry signs. The presented algorithm is called `EVSA` (eigenvector structure analysis) and constitutes an extension of the `SpecLoc2` algorithm described in our previous paper (Lucińska and Wierzchoń, 2014). While the algorithm is largely heuristic, it does seem to perform very well on a range of cases, reliably producing the expected number of clusters and giving reasonable results when applied to difficult real-world examples.

In Section 2, notation and standard spectral methods are presented, and the next section explains the main concepts used in the `EVSA` algorithm, which is presented in detail in Section 4. Then, in Section 5, we compare the performance of our algorithm with that of other solutions. Finally, in Section 6, the main conclusions are drawn.

2. Spectral methods

Spectral clustering methods have a strong connection with graph theory. The set of data points to be clustered will be denoted by $X = (x_1, x_2, \ldots, x_n)$. To each pair of points i and j, a symmetric similarity $s_{ij} \in [0,1]$ is attached. The value $s_{ij} > 0$ implies the existence of the undirected edge $i \sim j$ in the graph G spanned over the set of vertices X. The matrix $\mathbf{S} = [s_{ij}]$ plays the role of the adjacency matrix for G.

Let $d_i = \sum s_{ij}$ denote the degree of node i and let $\mathbf{D}$ be the diagonal matrix with d_i's on its diagonal. In an undirected and unweighted graph, the degree of a node is given by the number of its adjacent edges.

The set of edges between two subgraphs spanned over the sets $A, B \subseteq X$ is called the edge cut or a cut between A, B:

$$cut(A,B) = \sum_{x_i \in A, x_j \in B} s_{ij}. \tag{2}$$

Association, $assoc(A,X)$, of a set $A \subset X$ is the total weight connection from nodes in A to all nodes in the set X:

$$assoc(A,X) = \sum_{x_i \in A, x_j \in X} s_{ij}. \tag{3}$$

A clustering $\mathcal{C} = C_1, C_2, \ldots, C_K$ is a partitioning of X into the nonempty mutually disjoint subsets $C_1, C_2, \ldots, C_K$.

The Laplacian matrix associated with graph G is the $n \times n$ matrix $\mathbf{L} = \mathbf{D} - \mathbf{S}$. The normalized Laplacian is defined as $\mathbf{L}_n = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$. Since both $\mathbf{D}$ and $\mathbf{S}$

are symmetric real matrices, $\mathbf{L}$ and $\mathbf{L}_n$ are also symmetric and real, with non negative real-valued eigenvalues. If a graph is connected, the second smallest eigenvalue of its Laplacian is positive (Fiedler, 1975). Eigenvalues are usually sorted in ascending order in the case of Laplacians $\mathbf{L}$ and $\mathbf{L}_n$ and in descending order for $\mathbf{I} - \mathbf{L}_n$. The K dominant eigenvectors are eigenvectors associated with the K largest eigenvalues of $\mathbf{I} - \mathbf{L}_n$.

A partition of the nodes of a graph into K clusters is known as a K-way graph cut: in a special case when $K = 2$ we speak about bipartitioning. Spectral clustering algorithms can be classified according to two approaches: recursive spectral bipartitioning algorithms and direct K-way spectral clustering ones. The former solutions use the right eigenvector associated with the second smallest eigenvalue of the normalized or unnormalized Laplacian matrix, which is called the Fiedler vector. Divisions are recursively repeated until a K-way partition is found. Direct K-way methods use the first K eigenvectors and directly find partitions with the help of some heuristics.

The algorithm proposed by Shi and Malik (2000) is a well-known example of recursive methods. It aims at minimizing the normalized cut criterion proposed by the same authors. The normalized cut between two sets $A, B \subseteq X$ is defined as

$$Ncut(A,B) = cut(A,B)\Big(\frac{1}{assoc(A,X)} + \frac{1}{assoc(B,X)}\Big). \quad (4)$$

According to the authors, the set X is partitioned into two clusters $C, C' = X - C$ that minimize $Ncut(C, C')$ over all possible two way partitions of X. The algorithm consists of the steps shown in Algorithm 1.

Algorithm 1. Shi and Malik algorithm.

Input: Similarity matrix $\mathbf{S}$, number K of clusters to construct.

Output: Clusters $C_1, \ldots, C_K$

Step 1. Given a set of features, set up a weighted graph G and compute its similarity matrix $\mathbf{S}$ and its degree matrix $\mathbf{D}$.

Step 2. Solve a generalized eigenproblem $(\mathbf{D} - \mathbf{S})\mathbf{y} = \lambda\mathbf{D}\mathbf{y}$ for eigenvectors with the smallest eigenvalues.

Step 3. Use the eigenvector related to the second smallest eigenvalue ($\mathbf{y}_2$) to bipartition the graph by finding the splitting point such that $Ncut$ is minimized.

Step 4. Decide if the current partition should be subdivided and recursively repartition the segmented parts if necessary.

The number of groups segmented by this method is controlled directly by the minimum allowed $Ncut$. The second smallest Laplacian eigenvector $\mathbf{y}_2$ carries significant structural information regarding the connectivity of the graph. In an ideal case of two nonconnected subgraphs, $\mathbf{y}_2$ assumes just two discrete values, and the signs of the values can indicate exactly how to partition the graph. In a real case, the second eigenvector can take on continuous values and a splitting point must be chosen to cluster the components of $\mathbf{y}_2$. The authors suggested a few different ways of choosing a splitting point such as the zero or median values or finally the minimum $Ncut$, which is the approach they took in the presented algorithm. They checked a few evenly spaced possible splitting points, and computed the best $Ncut$ among them.

One of the most popular K-way algorithms is NJW, proposed by Ng *et al*. (2001) and shown in Algorithm 2.

Algorithm 2. NJW algorithm.

Input: Similarity matrix $\mathbf{S}$, number K of clusters to construct.

Output: Clusters $C_1, \ldots, C_K$

Step 1. Construct the complement of the normalized Laplacian matrix, $\mathbf{L}_c = \mathbf{I} - \mathbf{L}_n$.

Step 2. Find $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K$ largest eigenvectors of $\mathbf{L}_c$ and form the matrix $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K]$ by stacking the eigenvectors in columns.

Step 3. Form the matrix $\mathbf{F}$ from $\mathbf{Y}$ by renormalizing each of $\mathbf{Y}$'s rows to have unit length; in other words, project data onto a unit k-sphere.

Step 4. Treating each row of $\mathbf{F}$ as a point, cluster them into K clusters via K-means algorithm.

Step 5. Finally, assign the original point x_i to cluster j if and only if row i of the matrix F was assigned to cluster j.

The authors used a matrix perturbation theory argument to theoretically explain the algorithm. For a connected graph, the normalized Laplacian $\mathbf{L}_n$ is positive definite and has exactly one eigenvector with eigenvalue 1 and other eigenvalues smaller than 1. When the graph consists of K connected components, i.e., it can be perfectly separated into K non overlapping communities, the matrix $\mathbf{L}_c$ will be (with the rows ordered by clusters) block diagonal with K blocks. It will have exactly one eigenvector with eigenvalue 1 for each block. Therefore, each row in the matrix $\mathbf{F}$ will have exactly one entry equal to 1, all the others being zero and the data in this K-dimensional space will cluster on the unit length vectors on the coordinate axis. In more general cases, when the clusters are not strongly separated and the matrix $\mathbf{L}_c$ is not block diagonal, the validity of the algorithm is related to the existence of a large eigengap, i.e., the $(K+1)$-th eigenvalue of $\mathbf{L}_c$ needs to be significantly less than 1 to guarantee stability of the selected eigenspace under perturbations.

The eigengap heuristic is also a tool for establishing

the number of clusters. Here the goal is to choose the number K such that all eigenvalues $1, \ldots, K$ are close to 1 but $K+1$ is relatively small. In the case of weakly separated clusters the task is quite difficult, since the gap could be very small.

3. Properties of adjacency matrix spectra

3.1. Partitioning criteria based on the adjacency matrix. Recursive bipartitioning algorithms exploit the structure of the Laplacian's second smallest eigenvector only. On the other hand, K-way algorithms utilize K dominant eigenvectors, although they do not take into consideration the special structure properties of the eigenvectors, using usually the K-means algorithm for the final partitioning. In the EVSA algorithm, we applied the approach that utilizes ideas of both from the above described solutions.

Most spectral methods use eigenvectors and eigenvalues of different Laplacian matrices. Only a few of them take into consideration eigendecomposition of an adjacency matrix (e.g., Shi *et al*., 2009). The adjacency matrix of a simple graph presents in a straight way relations in a graph. As a diagonally constant, symmetric, and nonnegative matrix it has the eigenvectors and eigenvalues with some useful features, which will be presented below.

As mentioned in Section 2, Shi and Malik (2000) showed that the normalized Laplacian matrix leads to a normalized cut criterion. A similar argument can be set forth for an adjacency matrix.

Let us consider the Rayleigh quotient (Meyer, 2000) of the adjacency matrix $\mathbf{S}$ given as

$$R(\mathbf{S}, \mathbf{y}) = \frac{\mathbf{y}^T \mathbf{S} \mathbf{y}}{\mathbf{y}^T \mathbf{y}}, \tag{5}$$

where $\mathbf{y}$ is a nonzero vector.

According to the Courrant–Fischer theorem, the maximum of the Rayleigh quotient can be obtained as the largest eigenvalue $\lambda_{\max}$ of the adjacency matrix $\mathbf{S}$ and $\mathbf{y}_{\max}$ that yields this maximum is the corresponding eigenvector of $\lambda_{\max}$,

$$\max_{y \neq 0} R(\mathbf{S}, \mathbf{y}) = \lambda_{\max} = \frac{\mathbf{y}_{\max}^T \mathbf{S} \mathbf{y}_{\max}}{\mathbf{y}_{\max}^T \mathbf{y}_{\max}}. \tag{6}$$

If we replace $\mathbf{y}_{\max}$ with a characteristic function of a partition $A \subseteq X$, i.e., $\mathbf{y}_{\max}(i) = 1$ if the vertex x_i belongs to the cluster A and $\mathbf{y}_{\max}(i) = 0$ otherwise, then

$$\frac{\mathbf{y}_{\max}^T \mathbf{S} \mathbf{y}_{\max}}{\mathbf{y}_{\max}^T \mathbf{y}_{\max}} = \frac{\sum_{x_i \in A, x_j \in A} s_{ij}}{|A|}. \tag{7}$$

The right-hand part of the above equation equals the ratio association $Rassoc$ (Shi and Malik, 2000), which is another plausible criterion for partitioning, instead of the normalized cut criterion,

$$Rassoc(A, A) = \frac{assoc(A, A)}{|A|}, \tag{8}$$

where $|A|$ denotes the number of nodes in A.

Maximizing the ratio association leads to the largest within-cluster association relative to the size of the cluster. The eigenvector corresponding to the largest eigenvalue indicates the partition A with the maximum number of edges connecting nodes belonging to A.

By the Perron–Frobenius theorem (Meyer, 2000), since $\mathbf{S}$ is real, symmetric, and nonnegative, the entries of the eigenvector corresponding to the largest $\mathbf{S}$ eigenvalue are all nonzero and positive. This means that $\mathbf{y}_{\max}$ should be relaxed to take on real values, but it still includes the clustering information. Nodes with relatively large entries in $\mathbf{y}_{\max}$ belong to A, whereas those with entries close to zero belong to $B = X \setminus A$.

Maximization of the Rayleigh quotient for an adjacency matrix also allows us to distinguish partitions on the basis of the eigenvector sign. Let us assume that the entries of an indicator eigenvector $\mathbf{y}'$ equal 1 if a node belongs to A and -1 otherwise. The objective can be written as maximization, i.e.,

$$R(\mathbf{S}, \mathbf{y}'_{\max}) = \frac{\mathbf{y}'^T_{\max} \mathbf{S} \mathbf{y}'_{\max}}{\mathbf{y}'^T_{\max} \mathbf{y}'_{\max}}. \tag{9}$$

The numerator of the expression $\mathbf{y}'^T_{\max} \mathbf{S} \mathbf{y}'_{\max}$ takes the following form:

$$\max(assoc(A, A) + assoc(B, B) - 2cut(A, B)). \tag{10}$$

The maximum of the Rayleigh quotient for the matrix $\mathbf{S}$ and vector $\mathbf{y}'$ leads to the largest within-cluster association of A and B together with the smallest number of edges between the two clusters. The right part of the above expression constitutes a combined association/cut criterion, which reflects both intra-connectivity within the groups and inter-group linkage simultaneously.

As the eigenvector corresponding to the largest eigenvalue has only positive entries, the maximum of the Rayleigh quotient as a function of the vector $\mathbf{y}'$ equals the second largest eigenvalue of $\mathbf{S}$ (assuming A and B have some edges in common). The vector $\mathbf{y}'_{\max}$ is the second eigenvector and it will be used as the other indicator of nodes' membership. Similarly to the previous case, it should be relaxed to take on real values.

The first eigenvector corresponds to maximization of connectivity within a cluster, whereas the second one maximizes the combined criterion including both the association and cut. Using not only the largest but also the second largest eigenvector of a cluster, we gain additional information.

3.2. Case of multiple connected components. The Laplacian, normalized Laplacian, and adjacency matrices of a graph G, which consists of K connected components, are block diagonal and their eigenvalues and eigenvectors are the union of eigenvalues and eigenvectors of its blocks. In the case of Laplacians, any connected component possesses exactly one eigenvector which has a zero eigenvalue. If a graph has K connected components, then exactly one eigenvector of the first K eigenvectors of the Laplacian represents one component. This is not always true in the case of the adjacency matrix. It could happen that the two largest eigenvalues of the block diagonal adjacency matrix come from the same block and another block does not have its representation among the first K eigenvectors. Each cluster has its representation in the group of dominant adjacency matrix eigenvectors, but the group can include eigenvectors related to eigenvalues smaller then the K-th eigenvalue. For this reason, the adjacency matrix is not very often used in spectral clustering. We will show that the special properties of adjacency matrix eigenvalues and eigenvectors allow us to indicate the eigenvectors that reveal a graph structure.

The Perron–Frobenius theorem implies that if G is connected, then the largest eigenvalue $\lambda_{\max}$ of $\mathbf{S}(G)$ has multiplicity 1. This eigenvalue is a kind of a "graph average degree." More precisely, let $d_{\min}$ denote the minimum degree of G, let $\overline{d}$ be the average degree, and let $d_{\max}$ be the maximum degree. Lovász and Vesztergombi (2002) proved that, for every connected graph G,

$$\max\left(\overline{d}, \sqrt{d_{\max}}\right) \leq \lambda_{\max} \leq d_{\max}. \tag{11}$$

Moreover, Mohar (1989) showed that the second largest eigenvalue λ_2 of $\mathbf{S}(G)$ has an upper bound

$$\lambda_2 \leq \sqrt{d_{\max}^2 - h(G)^2}. \tag{12}$$

The constant $h(G)$ is called the isoperimetric number or modified Cheeger constant of G and is defined as

$$h(G) = \min_{\emptyset \neq C \subset X} \frac{|E(C, \overline{C}|}{\min(|C|, |\overline{C}|}, \tag{13}$$

where $E(C, \overline{C})$ are the edges connecting vertices in C with vertices in $\overline{C}$, the complement of C. The constant $h(G)$ can serve as a measure of the connectivity of graphs and the formula (12) is a Cheeger-type inequality similar to that introduced by Cheeger (1970). Graphs with low Cheeger constants are easier to cut, while those with high Cheeger constants are more robust to losing edges (Chung, 1997).

Brigham and Dutton (1984) found an upper estimate for the i-th eigenvalue of a graph as a function of the number of vertices and edges:

$$\lambda_i < \sqrt{2m\frac{n-i}{ni}}, \tag{14}$$

where m is the number of edges in the graph. This inequality is valid for $i = 1, \ldots, n$.

The above formulas allow us to explain some properties of the spectrum of the adjacency matrix of a graph G that consists of K connected components C_j. An eigenvector $\mathbf{y}(G)$ of $\mathbf{S}(G)$ representing a cluster C_j consists of two parts. One equals zero and includes entries corresponding to other clusters. The other equals the first eigenvector $\mathbf{y}_1(C_j)$ of the cluster C_j,

$$\mathbf{y}(G) = \begin{cases} 0 & \text{if } x_i \notin C_j, \\ \mathbf{y}_1(C_j) & \text{if } x_i \in C_j\,. \end{cases}$$

The eigenvector $\mathbf{y}_1(G)$ related to the largest eigenvalue $\lambda_1(G)$ of $\mathbf{S}(G)$ corresponds to the most compact cluster. Clusters that are sparsely connected are indicated by eigenvectors of $\mathbf{S}(G)$ related to smaller eigenvalues.

When a gap between the first and second eigenvalues, defined as the spectral gap,

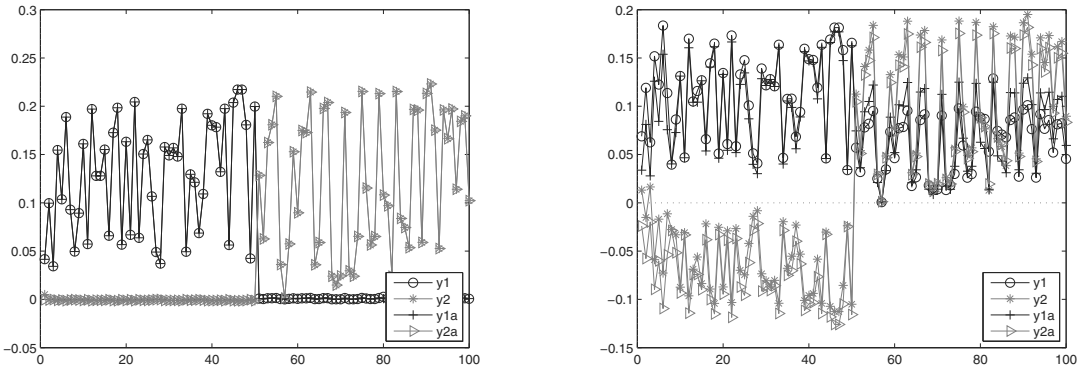
$$\delta_j = \lambda_1(C_j) - \lambda_2(C_j), \tag{15}$$

is large, $h(C_j)$ is large and C_j is quite compact. Moreover, if the density of the cluster C_j exceeds significantly the density of the others, a few of its eigenvalues $\lambda_i(C_j)$, where $i > 2$, can precede in the spectrum of G the first eigenvalue $\lambda_1(C_l)$ of the other cluster C_l, for $l \neq j$.

3.3. Perturbed eigenvalues and eigenvectors. The situation gets even more complicated when there are some edges between the nodes belonging to different clusters. We treat them as perturbation to an ideal case. Assume we are given an adjacency matrix $\mathbf{S}(G)$ with its K dominant eigenvalues $\lambda_1(G) \geq \lambda_2(G) \geq \cdots \geq \lambda_K(G)$ and its eigenvectors $\mathbf{y}_1(G), \mathbf{y}_2(G), \ldots, \mathbf{y}_K(G)$, respectively, and a perturbed matrix $\widetilde{\mathbf{S}}$. The matrix $\mathbf{S}$ is block diagonal, whereas the matrix $\widetilde{\mathbf{S}}$ contains also edges connecting different clusters. The perturbations are sufficiently small, that is, $\|\widetilde{\mathbf{S}} - \mathbf{S}\| < \varepsilon$ for some small ε, where $\|\mathbf{M}\|$ is the Frobenius norm defined as

$$\|\mathbf{M}\|^2 = \sum_{i,j} M_{i,j}^2. \tag{16}$$

In our case an adjacency matrix $\mathbf{S}$ relates to a graph consisting of K connected components. The matrix $\mathbf{\Delta} = \widetilde{\mathbf{S}} - \mathbf{S}$ represents nodes and edges connecting the subsets after perturbation of the ideal case. Moreover, only for the adjacency matrix the relation between $\mathbf{\Delta}$, $\widetilde{\mathbf{S}}$, and $\mathbf{S}$ is true, i.e., the unperturbed part $\mathbf{S}$ remains unchanged after adding perturbation. Contrary to the adjacency matrix, the block diagonal parts of the Laplacian matrix vary depending on the grade of each node and are influenced by perturbations. This is the reason why the adjacency



(a) (b)

Fig. 1. Two dominant eigenvectors of the perturbed adjacency matrix (y_1, y_2) and their approximations (y_{1a}, y_{2a}) for two well separated (a) and two weakly separated subgraphs (b).

matrix can generate better partitioning results. Since the adjacency matrix is symmetric and nonnegative, so is the perturbed matrix $\widetilde{\mathbf{S}}$.

The first-order approximation of the eigenpair $(\widetilde{\lambda}_i, \widetilde{\mathbf{y}}_i)$ of the perturbed matrix $\widetilde{\mathbf{S}}$ is given by the following equations (Stewart and Sun, 1990):

$$\widetilde{\lambda}_i \approx \lambda_i + \mathbf{y}_i^T \mathbf{\Delta} \mathbf{y}_i, \tag{17}$$

$$\widetilde{\mathbf{y}}_i \approx \mathbf{y}_i + \sum_{j \neq i} \frac{\mathbf{y}_j^T \mathbf{\Delta} \mathbf{y}_i}{\lambda_i - \lambda_j} \mathbf{y}_j. \tag{18}$$

The approximation error increases for eigenvectors that correspond to eigenvalues of smaller magnitudes. In other words, the quality and validity of the largest eigenpairs are not affected by the approximation. Since embedding into a lower dimensional space with the use of an adjacency matrix depends on a few largest eigenvectors, the approximation constitutes a faithful representation of the original source space. Taking the above into consideration, it is convenient to use another form of the perturbed eigenvector approximation given by Wu *et al*. (2011):

$$\widetilde{\mathbf{y}}_i \approx \mathbf{y}_i + \sum_{j \neq i}^{K} \frac{\mathbf{y}_j^T \mathbf{\Delta} \mathbf{y}_i}{\lambda_i - \lambda_j} \mathbf{y}_j + \frac{1}{\lambda_i} \mathbf{\Delta} \mathbf{y}_i. \tag{19}$$

The last formula is derived from the previous one with the use of the facts that $|\lambda_i| \gg |\lambda_j|$ for all $i = 1, \ldots, K$ (K: the number of clusters) and $\mathbf{y}_j^T \mathbf{\Delta} \mathbf{y}_i$ is just the projection of the vector $\mathbf{\Delta} \mathbf{y}_i$ onto one of the basis vectors $\mathbf{y}_j$.

We have extended this approximation to K dominant perturbed eigenvectors, taking into consideration entries of nodes having neighbors only in one subset. The condition $\|\mathbf{\Delta}\| < \varepsilon$ guarantees that the small number of eigenvector coordinates has been neglected by such simplification. Entries of nodes that are not directly connected with other subsets constitute the majority of all perturbed eigenvector entries and can be approximated by the following formula:

$$\widetilde{\mathbf{y}}_i \approx \mathbf{y}_i + \sum_{i \neq j}^{K} \frac{\mathbf{y}_j^T \mathbf{\Delta} \mathbf{y}_i}{\lambda_i - \lambda_j} \mathbf{y}_j. \tag{20}$$

Shmueli *et al*. (2012) analyzed the accuracy of the approximations given by the formulas (18)–(20). For the sake of simplicity, we will illustrate the formula (20) with an example. Consider a graph consisting of 100 nodes, creating two clusters (with an equal number of nodes), which have a few edges in common. The edges between the subsets make up a couple of percent of all graph edges. The adjacency matrix eigenvectors that correspond to the largest two eigenvalues according to our simplifications take the form of

$$\widetilde{\mathbf{y}_1} \approx \mathbf{y}_1 + \frac{\mathbf{y}_2^T \mathbf{\Delta} \mathbf{y}_1}{\lambda_1 - \lambda_2} \mathbf{y}_2, \tag{21}$$

$$\widetilde{\mathbf{y}_2} \approx \frac{\mathbf{y}_1^T \mathbf{\Delta} \mathbf{y}_2}{\lambda_2 - \lambda_1} \mathbf{y}_1 + \mathbf{y}_2. \tag{22}$$

Figure 1(a) presents two dominant eigenvectors of the perturbed graph adjacency matrix and their approximations given by (21) and (22). The number of edges between the subgraphs comprises about 1% of edges in one subset. Eigenvectors have nonzero entries at one subset of nodes and are close to zero at the rest of graph nodes. The difference between the unperturbed and perturbed eigenvector entries $|\widetilde{\mathbf{y}}_i| - |\mathbf{y}_i|$ is very small. The exact and the approximated values agree with each other. In Fig. 1(b) we can see the eigenvectors when the number of edges between subgraphs rises about four times. The entries of the first eigenvector are only a little higher at one of the subgraphs than at the other. The entries the second

eigenvector are negative for one subset and positive for the other, but their absolute values differ slightly between both the subgraphs. The approximation of the perturbed eigenvectors has become less accurate because the number of $\mathbf{\Delta}$ entries has increased.

We can use the above theoretical results to determine two indicators that provide information about in-group membership. First of all, we take into consideration absolute values of eigenvector entries. Nodes having the largest entry in one eigenvector belong to the same subset. The other important factor, determining division into subgraphs, is the sign of an eigenvector entry. It allows us to detect weakly separated clusters with common edges. Depending on the graph structure and eigenvector shape, it is therefore possible to use two different criteria to divide a graph.

We will explain our policy with the help of an example. Figure 2 presents the first, second, third, and fifth dominant eigenvectors of the Iris adjacency matrix. The adjacency matrix graph is constructed on the basis of the k-nearest neighbor metric with $k = 10$. The graph G consists of one well separated and two very close subgraphs. First we have to establish eigenvectors, representing different clusters, each with high entries within one cluster and small entries outside it. Figure 2 shows that the first dominant eigenvector takes values rather close to zero at nodes having large components in the second dominant eigenvector and vice versa. They therefore correspond to two different, completely disconnected subgraphs. The nonzero entries of each of them are proportional to the first eigenvector of the adjacency matrix related to the appropriate cluster. The first eigenvector represents the more dense subgraph (the number of edges $m = 1314$ and the subgraph average degree $\overline{d} = 26.28$) than the second one ($m = 1234$ and $\overline{d} = 12.34$).

Let us divide the graph into two subgraphs according

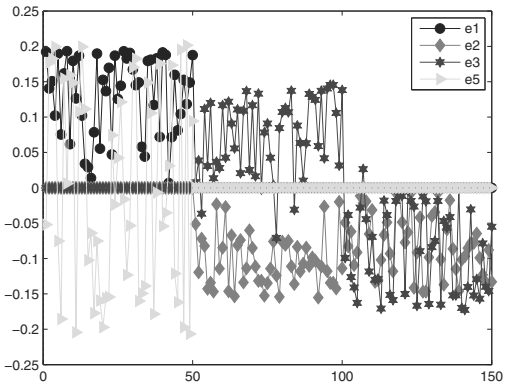


Fig. 2. First, second, third, and fifth dominant eigenvectors of the Iris dataset.

to the first and second eigenvector entries. The first 50 nodes belong to the subgraph A, whereas nodes from 51 to 150 create the subgraph B. If we calculate spectra for each subset separately, we can see that the second eigenvector $\mathbf{y}_2(A)$ of the subset A creates a nonzero part of the fifth eigenvector $\mathbf{y}_5(G)$ of the whole graph. Similarly, $\mathbf{y}_2(B)$ appears as a nonzero part of $\mathbf{y}_3(G)$ and $\mathbf{y}_3(B)$ as $\mathbf{y}_4(G)$ (the last one omitted in the picture for the sake of clarity). Do the third and fifth eigenvectors have different entry signs in different subgraphs? Should they both be used in partitioning?

According to (13), a large gap between the first and the second eigenvalue indicates a very compact subgraph. In the case of the subset A, not only the large spectral gap ($\lambda_1(A) = 24.635$ and $\lambda_2(A) = 14.291$) proves its high density, but also the fifth place of $\lambda_2(A)$ in the spectrum of the whole graph. Taking the above into consideration, we can draw the conclusion that partitioning according to $\mathbf{y}_5(G)$ signs would make no sense. Groups produced on the basis of the fifth eigenvector would have many edges in common. Things look quite different in the case of the subset B. Its first and second eigenvalues appear directly one after the other in the spectrum of the whole graph. The spectral gap between the eigenvalues is relatively small ($\lambda_1(B) = 20.617$ and $\lambda_2(B) = 19.491$). This means that the subset B has an internal structure and the signs of the entries of the third eigenvector represent different subgraphs.

Let us divide the subset B according to the sign of the nonzero part of $\mathbf{y}_3(G)$ into subsets $B1$ and $B2$ and then calculate mean absolute values of $\mathbf{y}_2(G)$ entries for both the subsets. The difference between the mean absolute values in the subsets is about 20 percent. Moreover, the mean absolute value of $\mathbf{y}_2(G)$ entries is higher in the subset $B1$, whereas the mean absolute value of $\mathbf{y}_3(G)$ entries is higher in the subset $B2$. Partitioning the subset A according to the sign of $\mathbf{y}_5(G)$ entries into two subsets $A1$ and $A2$ gives quite different results. First of all both $\mathbf{y}_1(G)$ and $\mathbf{y}_5(G)$ have higher mean absolute values in the subset $A1$ compared to the subset $A2$. Moreover, the mean absolute value of $\mathbf{y}_1(G)$ entries in the subset $A1$ exceeds only by about 7% the same value in the subset $A2$. The facts prove that the subset A as a compact subgraph can be divided only into subsets that have many edges in common, and matrix perturbation theory is not applicable in such a case.

The example illustrates also that, in order to find nodes belonging to the subset A, we have to compare the absolute values of the two dominant eigenvector entries. In the case of the weakly separated subsets $B1$ and $B2$, the difference between the eigenvector absolute values in the subsets is quite vague. Therefore, the above criterion can lead to many mistakes. Better results will be obtained on the basis of the $\mathbf{y}_3(G)$ entry sign.

Such an analysis of eigenvector properties allows us to establish the proper number of clusters. First, we have to find eigenvectors representing strongly separated subgraphs, i.e., with high entries within a group of nodes and small ones outside of it. Then the more difficult task is to indicate eigenvectors with different entry signs for two groups of nodes, allowing identification of weakly separated subgraphs. Having the right eigenvectors, we have the number of clusters. Of course, in some cases the decision whether partitioning on the basis of eigenvector entry signs leads to true group membership is ambiguous, similarly to visual judgments. As the choice of an eigenvector for clustering is made not only on the basis of its structure but also on the position of its corresponding eigenvalue in the spectrum, the partitioning depends on the properties of the whole dataset. The decision on whether or not a given subset should be divided is influenced by cohesion of other subsets.

3.4. Correlation between eigenvectors. In order to find eigenvectors representing different clusters, we calculated the correlation coefficient for each pair of the eigenvectors, taking the absolute values of the vector entries. The Pearson correlation coefficient between two vectors $\mathbf{y}_i$ and $\mathbf{y}_j$ is defined as

$$\rho(\mathbf{y}_i,\mathbf{y}_j)=\frac{\sum_{k=1}^{N}(y_{ik}-\overline{y_i})(y_{jk}-\overline{y_j})}{\sqrt{\sum_{k=1}^{N}(y_{ik}-\overline{y_i})^2}\sqrt{\sum_{k=1}^{N}(y_{jk}-\overline{y_j})^2}}. \tag{23}$$

The correlation coefficients $\rho(\mathbf{y}_i,\mathbf{y}_j)$ range from -1 to 1. If two vectors are linearly dependent, then the correlation between them will equal $+1$ or -1. The value of -1 indicates a perfect negative linear relationship between the vectors, and 0 means the lack of a linear relationship between the vectors. The correlation coefficient was chosen as an efficient indicator of whether or not two eigenvectors correspond to the same subgraph.

In Fig. 3 we can see absolute values of two simplified eigenvectors of a perturbed graph. We assume that both of them have only two entry values. The first eigenvector $\mathbf{y}_1$ is concentrated on parts I and III, and its values in this region equal $a1$ and are larger than the mean $\bar{y}_1$. Entries outside the subset are smaller than the mean and equal $b1$. In order to cover all possible relations between the values of two vectors, the second eigenvector is larger than its mean in parts II and IV and smaller for the rest of nodes. The eigenvectors are given by the following formulas:

$$\mathbf{y}_1 = a_1\hat{\mathbf{x}}_1 + b_1\hat{\mathbf{x}}_2 + b_1\hat{\mathbf{x}}_3 + a_1\hat{\mathbf{x}}_4, \tag{24}$$

$$\mathbf{y}_2 = b_2\hat{\mathbf{x}}_1 + a_2\hat{\mathbf{x}}_2 + b_2\hat{\mathbf{x}}_3 + a_2\hat{\mathbf{x}}_4, \tag{25}$$

where $a_i \geq \bar{y}_i > b_i$ for $i = 1, 2$, and $\hat{\mathbf{x}}_j$ equals 1 in n_j nodes of the j-th part and 0 in the remaining parts.

The mean values of the eigenvectors $\overline{y}_1$ and $\overline{y}_2$ equal

$$\bar{y}_1 = \frac{a_1(n_1+n_4)+b_1(n_2+n_3)}{n}, \tag{26}$$

$$\bar{y}_2 = \frac{a_2(n_2+n_4)+b_2(n_1+n_3)}{n}, \tag{27}$$

where n_j is the number of nodes belonging to the j-th part and $n = n_1 + n_2 + n_3 + n_4$ the number of nodes in the perturbed graph.

After splitting the sum into four parts and substituting $c_i = a_i - \bar{y}_i$ and $d_i = \bar{y}_i - b_i$ for $i = 1, 2$, the numerator of the correlation coefficient of $\mathbf{y}_1$ and $\mathbf{y}_2$ according to (23) equals

$$\begin{aligned}\operatorname{num}\rho(\mathbf{y}_1,\mathbf{y}_2) = &-c_1d_2n_1 - d_1c_2n_2 \\ &+ d_1d_2n_3 + c_1c_2n_4.\end{aligned} \tag{28}$$

Using (26) and (27), describing the mean values of $\mathbf{y}_1$ and $\mathbf{y}_2$, we have

$$c_1 = a_1 - \bar{y}_1 = \frac{(a_1-b_1)(n_2+n_3)}{n}, \tag{29}$$

$$d_1 = \bar{y}_1 - b_1 = \frac{(a_1-b_1)(n_1+n_4)}{n}. \tag{30}$$

From (29) and (30) we obtain

$$b_1 = \frac{a_1(n_1+n_4)}{n_2+n_3}, \tag{31}$$

$$b_2 = \frac{a_2(n_2+n_4)}{n_1+n_3}. \tag{32}$$

Substituting there results in (28), we get

$$\operatorname{num}\rho(\mathbf{y}_1,\mathbf{y}_2) = \frac{nc_1c_2(n_3n_4-n_1n_2)}{(n_1+n_3)(n_2+n_3)}. \tag{33}$$

As the denominator of the correlation coefficient is always nonnegative, $\rho(\mathbf{y}_1,\mathbf{y}_2)$ is negative if the product n_1n_2 is larger than n_3n_4 and positive otherwise.

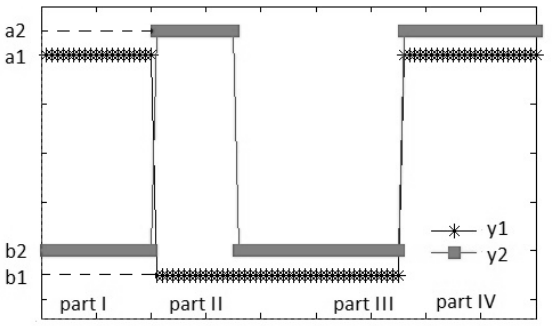


Fig. 3. Entries of two vectors presenting all possible relations between the dominant and another eigenvector of a connected graph.

If $\mathbf{y}_1$ and $\mathbf{y}_2$ are two most dominant eigenvectors, representing different clusters, the number of nodes in which they are both large (n_4) will be zero or very small. If the clusters do not have any edges in common, there will be no node with large entries in both eigenvectors. In such a case the product $n_3 n_4$ equals zero and the correlation coefficient is negative. A few edges connecting the clusters will result in a small value of n_4 and the value of the correlation coefficient will be close to zero.

Quite a different situation takes place if $\mathbf{y}_1$ is the most dominant eigenvector and $\mathbf{y}_2$ is the secondary eigenvector of the same subgraph. They are concentrated on the same subset, so the number of nodes in which both of them are large (n_4) and both of them take values smaller than their average (n_3) is significant. As eigenvectors corresponding to smaller eigenvalues are usually subject to a larger perturbation than the most dominant eigenvector, they can have large entries outside the subset considered (part II). On the other hand, they can be small in the regions where $\mathbf{y}_1$ is large (part I). Such a situations happen not very often, only in the case of nodes that have edges connecting two subgraphs. This means that the product $n_1 n_2$ is smaller than $n_3 n_4$, and the correlation coefficient is positive.

A similar relationship occurs not only for the first two but for all eigenvectors of a subgraph. However, the correlation coefficient is usually the largest one between the first pair of eigenvectors due to the increasing perturbation.

After calculating pairwise correlation coefficients for a few dominant eigenvectors of perturbed graphs, we are able to find both the eigenvectors connected with strongly separated subgraphs and the eigenvectors whose sign indicates weakly separated clusters. In the first case, the correlation coefficient between the eigenvector and the other eigenvectors, related to larger eigenvalues, is negative or close to zero. The secondary eigenvectors have a positive, rather large, correlation coefficient with only one preceding eigenvector.

4. Eigenvector structure analysis algorithm

The `EVSA` algorithm constitutes an extension of `SpecLoc2` (Lucińska and Wierzchoń, 2014), although some of their stages are executed in quite a different way. The main steps of our new solution are listed in Algorithm 3 and described in detail in the sequel. To create the matrix $\mathbf{S}$, the k-nearest neighbors were determined for each point x_i on the basis of the Euclidean metric. The adjacency matrix was obtained with the help of a mutual k-nearest neighbor graph. The graph was constructed by connecting x_i to x_j if x_i was among the k-nearest neighbors of x_j and vice versa. If two nodes are connected, the appropriate value in the adjacency matrix $\mathbf{S}$ equals 1, otherwise it is 0. The parameter k was chosen experimentally; however, there are some general heuristics which help to asses it. Brito *et al.* (1997) studied the relationship between the connectivity of a mutual k-nearest neighbor graph and the presence of a clustering structure and outliers in the data. They provided theoretic bounds on the value of k for which the mutual neighborhood graph remains disconnected and the connected components correspond to clusters.

Algorithm 3. `EVSA` algorithm.

Input: Adjacency matrix $\mathbf{S}$
Output: Clusters $C_1, \ldots, C_K$

Step 1. Find c dominant eigenvectors of $\mathbf{S}$.

Step 2. Calculate the pairwise correlation matrix of the dominant eigenvectors.

Step 3. Choose eigenvectors representing strongly separated subgraphs.

Step 4. Round the chosen eigenvectors.

Step 5. Partition nodes on the basis of the absolute values of the eigenvector entries.

Step 6. Choose eigenvectors having positive entries for one cluster and negative for others or vice versa.

Step 7. Partition nodes on the basis of the eigenvector entry sign.

Then we computed c dominant eigenvectors of the adjacency matrix. The number $c = 20$ was estimated as twice the maximum expected number of clusters, which guarantees a representation of each cluster by at least one eigenvector.

In order to find eigenvectors representing different clusters, we calculated the correlation coefficient for each pair of the c eigenvectors, taking the absolute values of the vector entries. The eigenvectors representing strongly separated subgraphs were chosen according to the discussion of the values of the correlation coefficient described in Section 3.4. These are eigenvectors that have a negative or close-to-zero correlation coefficient with others related to larger eigenvalues. The correlation coefficient between two eigenvectors representing two subgraphs is larger if the subgraphs have many edges in common than in the case of well separated subgraphs. In order to decide whether an eigenvector represents a subgraph, a dataset structure should be taken into consideration. We have introduced a parameter called the correlation threshold, which can take on one of the three values: 0.1, 0.2, and 0.3. In order to established the threshold value, the mean of the correlation coefficients between the first eigenvector and the other c dominant eigenvectors was calculated. The threshold value was chosen as a minimum that is larger than the mean correlation coefficient. We assumed that an eigenvector

$\mathbf{y}_u$ represents a strongly separated cluster if the following condition is met:

$$\rho(|\mathbf{y}_u|, |\mathbf{y}_j|) \leq tc, \tag{34}$$

where $j > u$, and tc is a correlation threshold. In our experiments we obtained the same results even if the values of the correlation threshold differed by a factor of about 0.2. Eigenvectors $\mathbf{y}_u$ form the set U.

According to Section 3.2, in the unperturbed case the first eigenvector of each connected component does not change the sign in nodes belonging to the component and has zero values in the remaining nodes. As each eigenvector from the set U is the first eigenvector of one subgraph, we rounded them in order to remove the same part of the perturbation. First we establish that of the eigenvector that equals the sign of its maximum absolute value. The sign of the maximum absolute value equals always $+1$ or -1 and never 0, because the eigenvectors are normalized. Then all entries with different signs were leveled to zero. In the rounded eigenvector there remain only entries of the same sign. The rounding procedure is shown in Algorithm 4.

Algorithm 4. Rounding procedure.

Input: Set U of eigenvectors representing strongly separated subgraphs
Output: Set UR of rounded eigenvectors from U
For each chosen eigenvector $\mathbf{y}_i \in U$:
Step 1. Find a node $x^i_{\max}$ with the maximum absolute entry in $\mathbf{y}_i$.

Step 2. Determine a set of nodes C_i that have a different sign in $\mathbf{y}_i$ than that of $x^i_{\max}$.

Step 3. In $\mathbf{y}_i$ assign zero value to nodes of C_i.

After the process of rounding, the chosen eigenvectors are used for partitioning nodes on the basis of absolute values. They are compared, and each point is given the label of the eigenvector in which it has the highest absolute value of the entry. In such a way all the nodes of the set are labeled, as all of them should be covered by nonzero parts of the rounded eigenvectors.

Each of the previously chosen eigenvectors corresponds to the first eigenvalue $\lambda_1(C_i)$ of $\mathbf{S}(C_i)$. In the next step of the algorithm, we checked whether each subgraph C_i can be divided into smaller subgraphs. If this is the case, the spectrum of $\mathbf{S}(C_i)$ and especially the second largest eigenvalue $\lambda_2(C_i)$ of $\mathbf{S}(C_i)$ and the eigenvector related to it should have special features, as described in Section 3. First, we checked whether the subgraph C_i is less dense than the other, already established, subgraphs C_j. Only not very compact subgraphs are subject to further division. Afterwards we examined the potential partitions to see if they comply with perturbation theory. Finally, we created a set UC containing selected eigenvectors. In the sixth step of the main algorithm the procedure shown in Algorithm 5 was executed.

Algorithm 5. Procedure of choosing eigenvectors with different signs in two clusters.

Input: Set of c dominant eigenvectors of $\mathbf{S}$ and set UR
Output: Set UC of eigenvectors having nonnegative entries for one cluster and negative for other or vice versa
Initialize $UC = \emptyset$.
For each eigenvector $\mathbf{y}_1(C_i) \in UR$:
Step 1. Choose a set of eigenvectors having their largest correlation coefficient with $\mathbf{y}_1(C_i)$.

Step 2. In the above set find the eigenvector related to the largest eigenvalue, i.e., $\lambda_2(C_i)$.

Step 3. Check whether the spectral gap of C_i is small compared with the other spectral gaps of $C_j \subset X$.

Step 4. Check whether $\lambda_l(C_i) < \lambda_1(C_j)$ for $j \neq i$ in the spectrum of $\mathbf{S}(X)$.

Step 5. Divide the subset, labeled by $\mathbf{y}_1(C_i)$ in the previous step of the main algorithm, into two subsets according to nonnegative and negative entries of $\mathbf{y}_2(C_i)$.

Step 6. Calculate mean absolute values of $\mathbf{y}_1(C_i)$ in both subsets.

Step 7. Check whether the difference between the mean absolute values is sufficiently large.

Step 8. Add the eigenvector $\mathbf{y}_2(C_i)$ to the set UC if the conditions 3, 4 and 7 are met.

The second eigenvector $\mathbf{y}_2(C_i)$ of $\mathbf{S}(C_i)$ is the first of the eigenvectors of $\mathbf{S}(X)$ that have the largest correlation with $\mathbf{y}_1(C_i)$. As shown in Section 3.4, eigenvectors concentrated on the same subgraph have high positive values of the correlation coefficient. According to our previous observations, the subset C_i can be divided if the spectral gap $\delta_i = \lambda_1(C_i) - \lambda_2(C_i)$ is relatively small (Section 3.2). We assumed that the spectral gap is small if it does not exceed half of the mean value of the other spectral gaps $\delta_j = \lambda_1(C_j) - \lambda_2(C_j)$, where C_j is the already established subset of X and $i \neq j$:

$$\delta_i < \frac{1}{2} \sum_j \frac{\delta_j}{|U|}. \tag{35}$$

According to the discussion in Section 3.2 the subgraph density exceeds significantly the densities of the other subgraphs of X when a few of its eigenvalues appear at the very beginning of the spectrum of $\mathbf{S}(X)$. In the algorithm we assumed that the subgraph having $l = 4$ subsequent eigenvalues at the very beginning of $\mathbf{S}(X)$ is much denser than the rest.

Further we considered the difference between the eigenvector mean absolute values in the subsets obtained

after the potential division. If it exceeded 20 percent, we assumed that the eigenvector $\mathbf{y}_2(C_i)$ distinguishes two separate subgraphs. As the eigenvector $\mathbf{y}_1(C_i)$ precedes the eigenvector $\mathbf{y}_2(C_i)$, its components are less perturbed than these of $\mathbf{y}_2(C_i)$. The difference between its mean absolute values in the two parts of the subgraph, obtained on the basis of the $\mathbf{y}_2(C_i)$ sign, should be more significant than for $\mathbf{y}_2(C_i)$ itself. This is a reason for comparing the mean values of $\mathbf{y}_1(C_i)$. Both the parameters, the number of subsequent eigenvalues and the difference between mean values, were chosen experimentally.

In the last step of the algorithm, the final partitioning on the basis of the eigenvector sign was applied to these subsets that have an internal structure. The division was made with the help of the eigenvectors from UC.

In our algorithm we make use of the first and secondary eigenvectors of each cluster. They were found thanks to the correlation coefficients calculated for each pair of eigenvectors. First, we chose eigenvectors with negative or close to zero correlations with the proceeding ones. They indicated strongly separated subgraphs. Next, for each first eigenvector the secondary one was found as positively correlated with the first. If some conditions were met, further partitioning was executed.

The main influence on the computational complexity of our algorithm has two steps: construction of the k-nearest neighbor graph and solution of the eigenproblem. The first step generates the complexity of computing distances between all pairs of datapoints ($O(n^2)$) and selecting k-nearest neighbors ($O(m + nlogn)$). To compute eigenvectors, we used the `eigs` function in MATLAB, which has a time complexity in $O(mch + nc^2h + c^3h)$, where m and n are respectively the numbers of edges and vertices of the graph, c is the number of eigenvectors to be computed, and h is the number of iterations for `eigs` to converge. Since $c < n$ and h is constant, the running time of `eigs` can be simplified to $O(mc + nc^2)$. Complexity of the other steps, like calculating the correlation matrix, finding valid parts of eigenvectors, establishing the set UC, and finally partitioning, does not exceed $O(nc^2)$). To sum up, the overall computational complexity of our algorithm is $O(mc + nc^2 + n\log n)$. The clustering problem can be solved in a reasonable amount of time for matrices of orders up to several thousands.

The computational complexity can be improved if other approximate methods of solving the eigenvalue problem are used instead of the algorithms implemented in MATLAB. Application of some Nyström methods (Lin *et al.*, 2015) guarantees linear time complexity $O(n)$.

5. Experimental tests

5.1. Benchmark algorithms and datasets. We compared the performance of the `EVSA` algorithm (implemented in MATLAB) with four other related clustering methods: the Ng, Jordan, and Weiss (`NJW`) algorithm (it was introduced in Section 2), the Zelnik-Manor and Perrona (`ZMP`) self-tuning spectral algorithm (the code available at `http://www.vision.caltech.edu/lihi/Demos/SelfTuningClustering.html`), the algorithm by Liu (`LTM-rd`) (the code available at `http://www.cse.ust.hk/~lkmpoon/`), and our previous solution `SpecLoc`. The first algorithm needs the number of clusters to be given as a parameter, whereas the others detect them automatically. The parameters needed for of the construction of the similarity matrix, i.e., the number of nearest neighbors k and the Gaussian scaling parameter σ, were established experimentally for each dataset. For each algorithm, k was chosen by testing many subsequent values. In the case of the `NJW`, `SpecLoc`, and `EVSA` algorithms, we constructed the similarity matrix on the basis on an unweighted graph. In order to tune the Gaussian scaling parameter for the `LTM-rd` algorithm, we systematically scanned a wide range of σ's (the range was set manually). The width of the Gaussian function in `ZMP` was adaptively assigned. As the presented solution constitutes an alternative to the K-means algorithm, we would like to show differences in their performance in exactly the same conditions. Therefore, the K-means algorithm was applied to the eigenvectors found by our solution.

First, we tested the algorithms with the help of very popular benchmark datasets from the UCI Machine Learning Repository (Dua and Karra Taniskidou, 2017). The smaller datasets include Iris, Breast cancer, Balance, Segment, Glass, Ecoli, and Yeast. Other examples from the UCI Machine Learning Repository are image datasets. The first one, a handwritten digit data set, Pendigit, consists of 250 samples from 44 writers. In our experiments the training and testing sets were used as two datasets, which are denoted by Pentra and Pentest, respectively. The second image dataset, LetterRec, contains the features of 26 capital letters of the English alphabet. The character images are based on 20 different fonts. They are randomly distorted. Another group consists of large data sets of handwritten digits, generated from the test dataset of MNIST (Yann and Corinna, 2009). It contains 10 handwritten digits with a total of 10,000 examples. Each example is a 28×28 gray-level image, and the dimensionality is 784. In addition, we constructed a few subsets of MNIST, namely, MNIST3 consisting of the examples of digits 3, 5, and 8, MNIST4 (digits 1, 2, 3, and 4), MNIST6 (digits 0, 3, 4, 6, 8, and 9), MNIST7 (digits 2, 3, 5, 6, 7, 8, and 9), MNIST8 (digits 0, 2, 3, 4, 5, 7, 8, and 9), MNIST9 (digits 1, 2, 3, 4, 5, 6, 7, 8, and 9). the US Postal Service (USPS) handwritten digit (Hull, 1994) contains ten digits 0–9, each sample being a 16x16 image. Two subsets of the USPS were also

Table 1. Summary of datasets.

Dataset	No. of instances	No. of features	No. of classes
Iris	150	4	3
Breast	683	9	2
Balance	625	4	3
Segment	2310	18	7
Glass	214	10	6
Ecoli	366	8	8
Yeast	1484	8	10
Pentest	3498	16	10
Pentra	7494	16	10
USPS	9298	256	10
USPS3	2357	256	3
USPS4	3919	256	4
MNIST3	2876	784	3
MNIST4	4159	784	4
MNIST6	5913	784	6
MNIST7	6903	784	7
MNIST8	7907	784	8
MNIST9	9020	784	9
MNIST	10000	784	10
Letters	20000	16	26

formed, which are denoted by USPS3 (digits 3, 5, and 8) and USPS4 (digits 1, 4, 7, and 9). The basic information on the datasets is summarized in Table 1. All the datsets are real-world examples, with the number of data points varying from 150 to 20,000, the number of dimensions from 4 to 784, and the number of groups from 2 to 26.

All the datasets are labeled, which enables evaluation of the clustering results against the labels using clustering accuracy (CA) and normalized mutual information (NMI) as popular measures of partition quality. For both measures a higher number means a better partitioning. We refer the interested reader to the work of Manning *et al.* (2008) for details regarding the measures. One should be aware, however, that specifying the true clustering is difficult, especially for real world data. Mostly the knowledge of domain experts is required and class-labels may not be adequate for the structure of the data or the evaluated cluster model.

5.2. Results. The performance of the algorithms is shown in Tables 2 and 3. We can see the superiority of the `EVSA` algorithm over the other tested solutions in the case of 70 percent of the datasets. The results illustrate that the presented method is competitive to the other solutions in terms of the quality of partitioning, measured with the help of NMI and clustering accuracy. For the algorithms `LTM-rnd` and `NJW`, the average values of CA and NMI as well as the standard deviations were computed on the basis of 20 runs. Furthermore, the best values among these 20 runs are reported. The other algorithms do not use any random methods. Standard deviations of their results are connected only with the accuracy of eigenvector computation and are smaller than 0.5 percent. There are no results for the Letters dataset in one case because it is too large for `ZMP` to run.

Apart from partitioning, our solution finds the number of clusters. In 9 out of 20 cases it was capable of indicating the right number of groups on the basis of eigenvalue and eigenvector analysis. Moreover, in 5 cases our solution failed to find only one cluster. The maximum difference between the true number and the number found by our method equals 3, whereas for the other algorithms the maximum difference is between 8 and 17. The mean difference between the true number of groups and the number found equals 1.05 for the `EVSA` algorithm, 3.4 for `LTM-rd`, 1.95 for `SpecLoc`, and 3.2 for the `ZMP` algorithm.

Clustering results of our solution did not depend on starting, randomly chosen settings, which is in sharp contrast to K-means clustering. Moreover, they were not very sensitive to the number of neighbors (the only parameter in our algorithm). Changes in the accuracy and NMI do not exceed 20% when the neighborhood size varied from 20 to 30 or 40 (Table 5).

6. Conclusions

The results show that the performance of our method is competitive with the other tested solutions not only in terms of the quality of clustering but also taking into

Table 2. Comparison of the EVSA, LTM-rnd, NJW, SpecLoc, and ZMP algorithms in terms of the NMI.

Dataset	EVSA	LTM-rd max	LTM-rd mean ± std	NJW max	NJW mean ± std	SpecLoc	ZMP
Iris	**0.89**	0.77	0.73 ± 0.08	0.81	0.78 ± 0.01	0.86	0.76
Breast	**0.80**	0.45	0.39 ± 0.02	0.80	0.75 ± 0.02	0.76	0.62
Balance	**0.62**	0.23	0.21 ± 0.04	0.30	0.29 ± 0.03	0.25	0.03
Segment	**0.68**	0.60	0.54 ± 0.03	0.57	0.55 ± 0.04	0.66	0.41
Glass	0.40	0.34	0.29 ± 0.05	0.43	0.41 ± 0.05	0.38	**0.44**
Ecoli	0.63	0.60	0.58 ± 0.03	0.61	0.59 ± 0.05	0.65	**0.68**
Yeast	**0.28**	0.27	0.23 ± 0.05	0.27	0.23 ± 0.04	0.25	0.22
Pentest	**0.86**	0.72	0.65 ± 0.07	0.80	0.75 ± 0.03	0.85	0.68
Pentra	**0.85**	0.82	0.73 ± 0.06	0.83	0.78 ± 0.04	0.84	0.80
USPS	0.79	**0.82**	0.79 ± 0.04	0.78	0.77 ± 0.03	0.76	0.40
USPS3	0.88	0.84	0.79 ± 0.06	**0.92**	0.88 ± 0.04	0.87	0.65
USPS4	0.91	**0.93**	0.87 ± 0.02	0.71	0.69 ± 0.02	0.90	0.66
MNIST3	**0.66**	0.62	0.58 ± 0.03	0.59	0.58 ± 0.03	0.65	0.15
MNIST4	**0.94**	0.87	0.85 ± 0.04	0.83	0.81 ± 0.04	0.90	0.73
MNIST6	**0.85**	0.84	0.79 ± 0.06	0.75	0.72 ± 0.04	0.79	0.59
MNIST7	**0.79**	0.78	0.71 ± 0.07	0.70	0.68 ± 0.05	0.62	0.34
MNIST8	0.76	0.71	0.68 ± 0.07	**0.79**	0.72 ± 0.06	0.61	0.38
MNIST9	**0.80**	0.74	0.71 ± 0.06	0.77	0.73 ± 0.05	0.70	0.32
MNIST	**0.80**	0.73	0.69 ± 0.09	0.70	0.67 ± 0.06	0.76	0.31
Letters	0.43	0.35	0.31 ± 0.08	0.42	0.39 ± 0.05	**0.44**	/

Table 3. Comparison of the EVSA, LTM-rnd, NJW, SpecLoc, and ZMP algorithms in terms of accuracy.

Dataset	EVSA	LTM-rd max	LTM-rd mean ± std	NJW max	NJW mean ± std	SpecLoc	ZMP
Iris	**0.97**	0.83	0.79 ± 0.03	0.93	0.91 ± 0.01	0.95	0.67
Breast-w	**0.97**	0.77	0.73 ± 0.02	**0.97**	0.95 ± 0.06	0.95	0.84
Balance	**0.89**	0.70	0.66 ± 0.05	0.60	0.59 ± 0.05	0.75	0.28
Segment	**0.71**	0.49	0.44 ± 0.02	0.48	0.45 ± 0.05	0.69	0.29
Glass	**0.53**	0.51	0.49 ± 0.03	0.44	0.37 ± 0.03	0.52	0.47
Ecoli	0.68	0.52	0.48 ± 0.03	0.72	0.65 ± 0.04	0.63	**0.77**
Yeast	**0.48**	0.41	0.30 ± 0.05	0.36	0.31 ± 0.04	0.46	0.41
Pentest	**0.95**	0.72	0.65 ± 0.07	0.71	0.68 ± 0.03	0.79	0.50
Pentra	0.82	0.67	0.62 ± 0.06	0.78	0.72 ± 0.03	**0.86**	0.80
USPS	0.73	0.71	0.68 ± 0.05	**0.77**	0.73 ± 0.07	0.70	0.30
USPS358	0.97	0.83	0.81 ± 0.03	**0.98**	0.91 ± 0.04	0.97	0.65
USPS1479	**0.98**	0.96	0.91 ± 0.03	0.63	0.61 ± 0.03	0.97	0.59
MNIST358	**0.88**	0.65	0.63 ± 0.04	0.83	0.79 ± 0.07	**0.88**	0.54
MNIST1234	**0.99**	0.79	0.75 ± 0.05	0.95	0.83 ± 0.09	0.97	0.71
MNIST034689	**0.94**	0.81	0.78 ± 0.04	0.71	0.61 ± 0.06	0.79	0.61
MNIST2356789	**0.88**	0.77	0.71 ± 0.09	0.78	0.74 ± 0.06	0.58	0.28
MNIST02345789	**0.86**	0.68	0.65 ± 0.08	0.62	0.58 ± 0.05	0.55	0.36
MNIST123456789	**0.81**	0.77	0.75 ± 0.07	0.65	0.59 ± 0.04	0.61	0.36
MNIST0123456789	**0.81**	0.71	0.66 ± 0.08	0.71	0.63 ± 0.04	0.72	0.22
Letters	0.31	0.30	0.28 ± 0.09	0.32	0.30 ± 0.03	**0.33**	/

consideration the determined number of clusters. The special features of the adjacency matrix allow us to obtain more exact results in the case of our algorithm. As the changes after adding perturbation in the adjacency matrix structure are smaller than for the other matrices, the adjacency matrix eigenvalues and eigenvectors give clearer picture of partitioned data. However, in a few cases the performance of our solution was poorer than that of the

Table 4. Comparison of the EVSA, LTM-rnd, SpecLoc, and ZMP algorithms in terms of the number of clusters.

Dataset	true	EVSA	LTM-rd	SpecLoc	ZMP
Iris	3	3	3	3	2
Breast	2	2	12	2	5
Balance	3	2	8	2	6
Segment	7	9	10	17	4
Glass	6	5	8	5	5
Ecoli	8	5	9	4	4
Yeast	10	9	12	6	6
Pentest	10	12	8	10	5
Pentra	10	13	11	10	10
USPS	10	9	14	7	2
USPS3	3	3	4	3	2
USPS4	4	4	5	4	4
MNIST3	3	3	7	3	2
MNIST4	4	4	5	4	5
MNIST6	6	6	7	4	4
MNIST7	7	7	10	5	2
MNIST8	8	8	11	5	3
MNIST9	9	8	15	6	3
MNIST	10	8	11	8	2
Letters	26	29	9	30	/

Table 5. Influence of the number of mutual nearest neighbors on EVSA performance.

Dataset	k range	NMI max	NMI mean $\pm$ std	CA max	CA mean $\pm$ std
Balance	20–30	0.62	0.53 ± 0.06	0.89	0.86 ± 0.02
Segment	20–40	0.68	0.64 ± 0.01	0.71	0.57 ± 0.03
Pentest	20–40	0.86	0.83 ± 0.03	0.95	0.77 ± 0.07

other algorithms. This can be explained by the fact that for all the datasets we used a very simple metric, contrary to the other cases.

Our algorithm was capable of finding the number of clusters more accurately than the other presented methods. Moreover, our solution was faster than the LTM-rd and ZMP algorithms, because it does not involve any iterative procedure to accomplish the task. In addition, the determination of the number of clusters does not require application of any quality function, contrary to the other presented solutions. The EVSA algorithm uses only properties of eigenvalues and eigenvectors.

Although our method at some points resembles that of Ng, Jordan, and Weiss, we added novel elements which have significant influence on algorithm performance. First of all, we gave up partitioning on the basis of a few dominant eigenvectors but chose eigenvectors that carry important information about the dataset structure. In some cases, subgraph densities differ strongly between themselves and a few eigenvalues of one subgraph can appear subsequently in the spectrum of the whole graph. Our approach allows us to find eigenvectors related to less dense clusters. We took into consideration not only eigenvectors related to well separated subgraphs, but also to those with common edges.

As can be seen from Tables 2 and 3 in Section 5.2, partitioning on the basis of exploiting special features of eigenvectors is more accurate than that performed with the help of the K-means algorithm. After applying both methods to the same eigenvectors, we obtained better results for our solution. In the case of weakly separated clusters, entries of nodes belonging to different groups are sometimes too similar to be distinguished by K-means algorithm. Our method compares eigenvector entries using two independent criteria, which results in a better quality of partitioning. The striking differences in results of the two algorithms show that our approach may be complimentary to the K-means (widely used in spectral clustering) and serve in situations, where the latter fails.

We presented a new spectral clustering algorithm that is a modification of our previous work. It uses eigenvectors of an adjacency matrix, obtained on the basis of a very simple metric. Its novelty lies in making use of eigenvalue and eigenvector properties, which result from matrix perturbation theory and graph theory. Our experiments showed the advantage of the EVSA algorithm

over both the traditional and new spectral clustering methods. This is because our solution is based on properly chosen eigenvectors. Moreover, it exploits some differences between node entries, which are too vague for K-means algorithm to be detected.

References

Azran, A. and Ghahramani, Z. (2006). Spectral methods for automatic multiscale data clustering, *in* A. Fitzgibbon *et al.* (Eds.), *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Washington, DC, pp. 190–197.

Dua, D. and Karra Taniskidou, E. (2017). *UCI Machine Learning Repository*, University of California, Irvine, CA, `http://archive.ics.uci.edu/ml`.

Brigham, R. and Dutton, R. (1984). Bounds on graph spectra, *Journal of Combinatorial Theory: Series B* **37**(3): 228–234.

Brito, M., Chavez, E., Quiroz, A. and Yukich, J. (1997). Connectivity of the mutual k-nearest-neighbor graph, *Statistics and Probability Letters* **35**(1): 33–42.

Cheeger, J. (1970). A lower bound for the smallest eigenvalue of the Laplacian, *in* R. Gunning (Ed.), *Problems in Analysis*, Princeton University, Princeton, NJ, pp. 195–199.

Chung, F. (1997). *Spectral Graph Theory*, American Mathematical Society, Providence, RI.

Fiedler, M. (1975). A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory, *Czechoslovak Mathematical Journal* **25**(4): 619–633.

Hull, J. (1994). A database for handwritten text recognition research, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **16**(5): 550–554.

Jia, H., Ding, S., Xu, X. and Nie, R. (2014). The latest research progress on spectral clustering, *Neural Computing and Applications* **124**(7–8): 1477–1486.

Lin, M., Wang, F. and Zhang, C. (2015). Large-scale eigenvector approximation, *Pattern Recognition* **48**(5): 1904–1912.

Liu, A., Poon, L., Liu, T.F. and Zhang, N. (2014). Latent tree models for rounding in spectral clustering, *Neurocomputing* **144**: 448–462.

Lovász, L. and Vesztergombi, K. (2002). Geometric representations of graphs, *in* G. Halász *et al.* (Eds.), *Paul Erdős and His Mathematics*, Bolyai Society Mathematical Studies, Budapest, pp. 471–498.

Lucińska, M. and Wierzchoń, S.T. (2014). Spectral clustering based on analysis of eigenvector properties, *in* K. Saeed and V. Snasel (Eds.), *Proceedings of Computer Information Systems and Industrial Management Applications*, Springer, Berlin/Heidelberg, pp. 43–54.

Manning, C., Raghavan, P. and Schütze, H. (2008). *An Introduction to Information Retrieval*, 1st Edn., Cambridge University Press, Cambridge.

Meyer, C. (2000). *Matrix Analysis and Applied Linear Algebra*, 1st Edn., SIAM, Philadelphia, PA.

Mohar, B. (1989). Isoperimetric numbers of graphs, *Journal of Combinatorial Theory* **47**(3): 274–291.

Ng, A., Jordan, M. and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm, *in* T. Dietterich *et al.* (Eds.), *Advances in Neural Information Processing Systems*, MIT Press, Cambridge, MA, pp. 849–856.

Schwarz, G. (1978). Estimating the dimension of a model, *Annals of Statistics* **6**(2): 460–464.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(8): 888–905.

Shi, T., Belkin, M. and Yu, B. (2009). Data spectroscopy: Eigenspaces of convolution operators and clustering, *Annals of Statistics* **37**(6b): 3960–3984.

Shmueli, Y., Wolf, G. and Averbuch, A. (2012). Updating kernel methods in spectral decomposition by affinity perturbations, *Linear Algebra and Its Applications* **437**(6): 1356–1365.

Stewart, G.W. and Sun, J. (1990). *Matrix Perturbation Theory*, 1st Edn., Academic Press, New York, NY.

von Luxburg, U. (2007). A tutorial on spectral clustering, *Statistics and Computing* **17**(4): 395–416.

Wu, L., Ying, X., Wu, X. and Zhou, Z.-H. (2011). Line orthogonality in adjacency eigenspace with application to community partition, *in* T. Walsh (Ed.), *Proceedings of the 22nd International Joint Conference on Artificial Intelligence,* AAAI Press, Menlo Park, CA, pp. 2349–2354.

Xiang, T. and Gong, S. (2008). Spectral clustering with eigenvector selection, *Pattern Recognition* **41**(3): 1012–1029.

Yann, L. and Corinna, C. (2009). *The MNIST Database of Handwritten Digits*, `http://yann.lecun.com/exdb/mnist/`.

Zelnik-Manor, L. and Perona, P. (2004). Self-tuning spectral clustering, *in* L.K. Saul *et al.* (Eds.), *Advances in Neural Information Processing Systems*, MIT Press, Cambridge, MA, pp. 1601–1608.

Zheng, X. and Lin, X. (2004). Automatic determination of intrinsic cluster number family in spectral clustering using random walk on graph, *Proceedings of the International Conference on Image Processing, Singapore, Singapore*, pp. 3471–3474.

Małgorzata Lucińska received the MSc degree from Jagiellonian University, Kraków, Poland, in 1989, and the PhD degree from the Systems Research Institute, Polish Academy of Sciences, in 2006. She works at the Kielce University of Technology, Poland. Her research interests include computational intelligence, machine learning, pattern recognition, spectral graph theory, and biologically inspired computations.

Sławomir T. Wierzchoń received the MSc and PhD degrees in computer science from the Technical University of Warsaw, Poland. He holds a DSc degree in computer science from the Polish Academy of Science. In 2003 he received the title of a professor from the President

of the Republic of Poland. Currently he is a full professor at the Institute of Computer Science of the Polish Academy of Sciences. He has published over 100 peer reviewed papers in various international journals and conferences, and 11 books in the field of machine learning. He has cooperated with medical centers in the area of statistical analysis and knowledge discovery in databases, and has participated in national and international research projects concerning various topics in the area of machine learning.

Received: 2 October 2017
Revised: 19 May 2018
Accepted: 10 June 2018