SOM-empowered Graph Segmentation for Fast Automatic Clustering of Large and Complex Data

Erzsébet Merényi, senior member
Department of Statistics and
Department of Electrical and Computer Engineering
Rice University, Houston, Texas 77005
Email: erzsebet@rice.edu

Joshua Taylor, student member
Department of Statistics
Rice University
Houston, Texas 77005
Email: jtay@rice.edu

***Abstract*—Many clustering methods, including modern graph segmentation algorithms, run into limitations when encountering “Big Data”, data with high feature dimensions, large volume, and complex structure. SOM-based clustering has been demonstrated to accurately capture many clusters of widely varying statistical properties in such data. While a number of automated SOM segmentations have been put forward, the best identifications of complex cluster structures to date are those performed interactively from informative visualizations of the learned SOM’s knowledge. This does not scale for Big Data, large archives or near-real time analyses for fast decision-making. We present a new automated approach to SOM-segmentation which closely approximates the precision of the interactive method for complicated data, and at the same time is very fast and memory-efficient. We achieve this by infusing SOM knowledge into leading graph segmentation algorithms which, by themselves, produce extremely poor results segmenting the SOM prototypes. We use the SOM prototypes as input vectors and CONN similarity measure, derived from the SOM’s knowledge of the data connectivity, as edge weighting to the graph segmentation algorithms. We demonstrate the effectiveness on synthetic data and on real spectral imagery.**

I. Introduction

A. SOM-clustering for Complex Data

Self-Organizing Maps (SOMs, Kohonen [1]) have been shown superior to many other methods in clustering highly structured manifolds (data with complex, irregular and noisy cluster structure, high feature dimension n, and / or large volume, Merényi et al. [2]). This makes SOMs prime candidates for making discoveries in Big Data scenarios.

Finding clusters with SOMs is a two-stage process. First, an SOM, which consists of a rigid (usually 2-dimensional; 2-D from hereon) lattice of neurons $i, i = 1, \cdots, P$, each with a prototype (weight vector) $w_i \in \mathcal{R}^n$ attached to it, learns the structure of a given data manifold $M \subset \mathcal{R}^n$ comprising N samples $\{\mathbf{x}_k\}_{k=1}^N$, $\mathbf{x}_k = (x_{k1}, \cdots, x_{kn}) \in M$, typically $N \gg P$. This is achieved through iterative adaptation of the prototypes to follow the data distribution. Simultaneously, the prototypes are organized on the SOM lattice in a topology-preserving fashion. Conditional on correct learning, the prototypes provide faithful approximation of the data distribution and their topological ordering on the lattice reflects their similarity relationships in the data space. Tools to assess the correctness of learning such as topology preservation measures (Villmann et al. [3], de Bodt et al. [4], Zhang and Merényi [5]) exist and should be applied before cluster extraction, but we omit an explanation here for space considerations. See an overview and references in Merényi et al. [2]. Further information that can be derived from a learned SOM and visualized include the number of data points mapped to each prototype (the size of the receptive field, RF_i of prototype w_i, visualized as a “hit map”), the data space distances of lattice-neighbor prototypes (U-matrix and variants, Ultsch [6], Hamel and Brown [7], octagonal erosion by Cottrell and de Bodt [8], mU-matix, Merényi et al. [9]), or more involved quantities like connectivity of prototypes, $CONN$ by Taşdemir and Merényi [10]. These can be used for locating contiguous groups of similar prototypes in the SOM grid where prototypes in each group collectively represent a cluster of similar data points.

However, while clusters may readily emerge from such visualizations in relatively simple cases, cluster extraction from highly structured manifolds is challenging because the visualizations become much less clear-cut (Merényi et al. [2]). Yet, it is in such cases where interactive segmentation tends to produce better quality than automated methods, but it requires expertese and can be time-consuming. This does not scale with the demands of near-real time processing, autonomous situations or large archives, where it is most needed.

B. Objectives of this Work

We present a new automated approach to SOM segmentation which closely approximates the precision of the interactive counterpart for complicated data and, at the same time, is very fast and memory-efficient. We achieve this by infusing SOM knowledge into leading graph segmentation algorithms which, by themselves, produce extremely poor results from the SOM prototypes alone. The break-through comes when, in addition to using the SOM prototypes, we also provide an SOM-derived similarity measure for the graph-cutting algorithms. We evaluate our approach with five graph segmentation methods on synthetic and real spectral imagery, by qualitative measures against known or previously verified clusterings. This extends our work in Merényi et al. [11] both in breadth and depth, as further elaborated in the Discussion (Section IV).

C. Previous Work in Automation of SOM-segmentation

Several automated approaches segment the SOM prototypes via hierarchical agglomerative clustering (HAC). HAC is favored over parametric or partitive methods because it can handle high-D inputs and irregularly shaped clusters (with an appropriate distance metric). Various Euclidean distance-based linkage metrics between pairs of prototypes such as centroid linkage (Vesanto and Alhoniemi, [12]), Ward measure (Cottrell and Rousset [13]) or centroid linkage constrained by grid neighborhood contiguity (Murtagh [14]) have been demonstrated to work well in HAC for relatively low-D data with a small number (3–8) of clusters. Cottrell and Rousset [13] captures more than 30 clusters from 12-D time series data but give no evaluation about how these clusters match relevant groupings as their focus is to illuminate the power of the SOM representation for interpretation. While HAC allows detection of variable cluster shapes the use of centroid linkages imposes a limitation to spherical clusters. Brugger et al. [15] propose combining the pairwise distances of lattice neighbor prototypes with the winning frequencies. This is the same information as contained in the mU-matrix (Merényi et al. [9]) from which the authors generate a smooth function called the Clusot surface via mixtures of "modified" Gaussians whose standard deviations are direction-dependent and computed from the (normalized) data-space distances to neighbors in the respective directions in the lattice, weighted by the winning frequencies. The resulting Clusot surface has valleys where large prototype distances coincide with low winning frequencies. The Clusot surface (more precisely, its graph representation) is then subjected to recursive flooding to detect mountain peaks above a flood line as clusters. Experimental results are modest, probably owing to parametrization problems such as edge weights in the graph. Common to the above examples, the data sets are small (a few thousands of samples), and in most cases contain few clusters (3–7). SOMs of larger data sets — but still of low dimensionality and complexity — are segmented with HAC and restricted connectivity by Goncalves et al. [16] capturing four to five clusters of land cover from IKONOS and Landsat5 3-band imagery with very high (92 – 99%) accuracy. They restrict the merging to lattice neighbor prototypes and exclude, from the HAC phase, dead and heterogeneous prototypes from transition regions between clusters, which are identified by high relative dispersion of the pixel features in their receptive field.

More complex spectral imagery is segmented by Taşdemir et al. [17]. Four million 20-D spectral signatures are learned with a 50 x 50 SOM, which is then clustered with HAC methods to compare the effects of Ward, centroid, average, and $CONN$ linkages. The HAC with $CONN$ linkage generally outperforms the others as well as k-means clustering, in detecting ten desirable clusters. The $CONN$ linkage is derived from the $CONN$ similarity measure of prototypes defined in Taşdemir and Merényi [10]. Since our automated approach also relies on this measure we review $CONN$ briefly below.

Liao et al. [18] apply HAC to SOMs of fMRI data using a novel spatio-temporal distance measure composed of pairwise correlations of prototypes weighted by an exponentially decaying function of their lattice distance. While the correlation measure admits varied cluster shapes, the combined measure may fail at sharp boundaries where backward topology violations occur and outline strong cluster separation (*e.g.,* high "fences" in the mU-matrix). The relatively large weighting generated by close lattice proximity of two prototypes across such a cluster boundary may counteract the small correlation of those prototypes in data space. This could explain why their cluster identification success is limited to 3–4 brain areas.

As an alternative to HAC, Taşdemir [19] explores Spectral Clustering (SC) of the graph Laplacian of SOM prototypes. SC (with scale parameter local to prototypes) outperforms HAC methods with the above linkages in finding eight true clusters in a remote sensing spectral image cube with 41 bands (input features) and 216,000 samples. Since no information is given on the cluster sizes or their spectral signatures it remains unknown if these clusters are well-separable, uniform in size, etc. The average accuracies, computed from 50 repeated runs with each method, are also close in many cases making the conclusion unclear without knowing the standard deviations of the runs. Interestingly, the same SC method performs significantly poorer on seemingly simple 2-D data sets with 2–3 clusters. The reason may be that these data sets have specific challenges such as variable cluster shapes and densities vs. proximities, whereas other data (from the UCI repository) may have clusters more balanced in size and other properties.

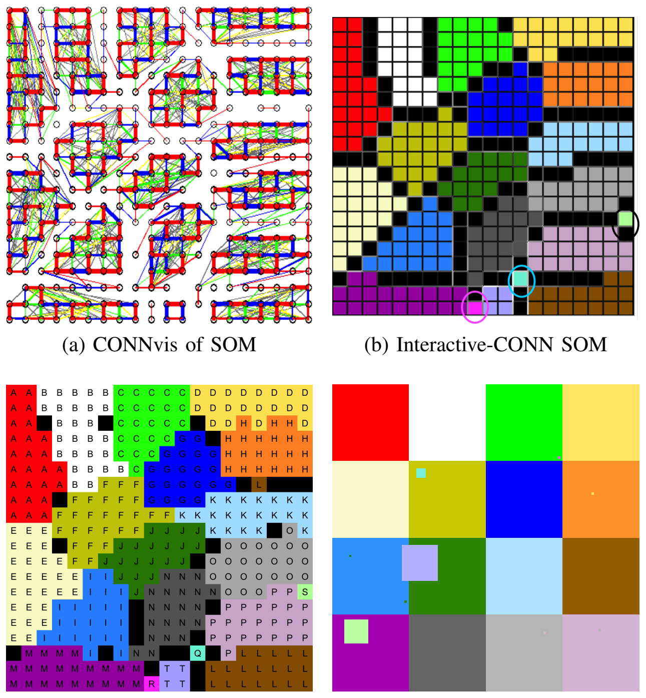
The above also underlines that while clustering difficulties can be caused by size, dimensionality, the number of inherent partitions and noise, the complexity of a manifold — and thus the clustering challenge — ultimately depends on the variations in cluster sizes, shapes, densities, and the number and relative positions of clusters. We aim to address the combination of these challenges illuminated by previous work.

D. The $CONN$ Similarity Measure for SOM Segmentation

For capturing complex cluster structure interactively from SOMs we have successfully used the $CONN$ similarity measure, which is derived from the converged SOM and expresses manifold connectivity rather than data space distances (Taşdemir and Merényi [10], [20]). The connection strength $CONN(i, j)$ between prototypes attached to neurons i and j is measured, during a full recall on the data set, as the number of data vectors which choose one prototype as their SOM winner and the other as second winner. The $CONN$ matrix can be visualized over the SOM to guide cluster extraction. This is illustrated in Fig. 1 through the SOM segmentation of the 6-D 20-class synthetic "spectral" image cube described in Section III-A. (In spectral images, n-D feature vectors are attached to (x, y) spatial locations; these are the input vectors to SOM learning.) In Fig. 1 (a) the SOM lattice of 20 x 20 neurons (black dots) is shown with the $CONN$ representation of the learned manifold structure. A cell with no dot has an empty prototype. In addition to the thickness of the line segments, which expresses global relations of the relative connection strengths, colors indicate the relative importance — a local ranking — of the connections to other prototypes, in the order of red (most-connected), blue, green, yellow, and grey shades. Together, the global connection strengths and local rankings provide rich information about where the manifold is strongly woven and where it is disconnected or thin.

The $CONN$ representation also reveals topology violations. Martinetz and Schulten [21] proves that (under mild conditions) two prototypes get connected if they are Voronoi neighbors in data space. Thus perfect topology preservation is achieved when prototypes are connected to their lattice neighbors, or — in case of a disconnected submanifold — there is no connection to lattice neighbors at cluster boundaries (these signify *backward* topology violations, which are helpful in finding clusters). Line segments connecting prototypes with a lattice distance larger than one indicate *forward* topology violations, and the line length and width, respectively, express the *extent* and the *severity* of the violation. These can be analyzed to separate serious violations from those inconsequential for capturing clusters. Intuitively, we see from Fig. 1 (a) that clusters have many relatively weak violating intra-cluster connections which do not interfere with cluster identification, while connections across clusters are weak or missing. $CONN$ informs about the mapping quality, both visually (with $CONN$vis overlay) and through quantitative $CONN$-based topology measures (see Merényi et al. [2]).

Cluster boundaries are found between regions that are strongly connected inside and have thin or no connections to other regions. For visualization, a non-linear binning of $CONN$ strengths is applied to aid the human eye. The bin boundaries are automatically derived from $CONN$ statistics. Details on this and a cluster extraction procedure are given in Taşdemir and Merényi [10]. Fig. 1 (b) shows the SOM lattice with clusters of similar prototypes extracted interactively from the $CONN$vis representation. On Fig. 1 (c) each prototype is marked by the majority class label (of the samples mapped to it). Slight differences with Fig. 1 (b) are caused by a few samples mapping to prototypes not in their classes (due to noise, or imperfect SOM learning at cluster boundaries). More importantly, a few non-empty prototypes were left unlabeled by the human analyst. These have only 1–2 samples mapped to them causing negligible omission or confusion as seen in Fig. 1 (d) which shows the clusters in data space. This clustering is almost perfect match to the true classes.


(a) CONNvis of SOM (b) Interactive-CONN SOM

(c) Majority labels on SOM (d) Interactive-CONN image

Fig. 1. (a) Visualization of $CONN$ values ($CONN$vis) over the 20x20 SOM of the 6-D 20-class data cube described in Section III-A. It is easy to discern at least 18 of the classes. (b) Interactively extracted clusters labeled (by different colors) on the SOM. Ovals highlight rare clusters. (c) Majority labels of samples mapped to each prototype. Letter labels (redundant with colors) are also indicated. Differences with Fig. 1 (b) occur because a few samples map to prototypes in other classes (due to noise, or imperfect SOM learning at cluster boundaries), or the human analyst did not label some prototypes in (b). The "offending" prototypes have 1–2 samples mapped to them, which accounts for the negligible confusion in image space. (d) Clusters shown in the spatial image. This matches the spatial layout of the true data clusters except for a few stray pixels that resulted from imperfect SOM learning.

II. Graph Segmentation Methods and Their Use for SOM Clustering

Graph-based community detection, also known as graph segmentation, aims to find communities of graph vertices connected by edges which represent community membership. Formally, a graph $\mathcal{G}$ is a collection of N vertices $\mathcal{V}$ and a binary $N \times N$ adjacency matrix A whose (i, j)-th entry represents connectivity between vertices i and j. An extension of this is a weighted graph, where A is replaced by matrix E where E_{ij} denotes a graded similarity measure between vertices i and j. Data clustering can then be achieved through graph segmentation algorithms, where each vertex represents an observation and each edge represents some prescribed similarity measure between vertices. These algorithms all produce an optimal partitioning $C^* = \{C_1, \dots, C_K\}$ of $\mathcal{V}$ into K mutually exclusive sets representing clusters of data points.

A. Background on Algorithm Classes

Graph segmentation algorithms all generally begin with the same goal: given a partitioning C, define some measure $Q(C)$ indicating the quality of the partitioning (in some sense) and optimize it with respect to C. Approaches to this problem from the fields of computer science, physics and statistics have resulted in different characterizations of Q and an array of procedures tailored to its optimization (see Fortunato [22] for a thorough overview of algorithm classes). We have experimented with several of these methods; the best performers for our purposes here are highlighted below.

The Fast & Greedy algorithm (Clauset et al. [23]) is an extension of the modularity based algorithm of Newman [24], fine tuned for computational performance. *Modularity* is a popular quality measure of a partitioning which seeks to capture the *relative* importance of intra-cluster strength (i.e., weighted connections) compared to what would be expected from random partitioning, subject to certain constraints. Thus, it is concerned not with how strongly (in absolute terms) components are clustered together, but rather with how significant such an observed strength is relative to what could be expected by chance. Given a partitioning of vertices $C = \{C_1, \dots, C_K\}$, let $\beta = ||\text{vec}(E)||_1$, so β is (double) the sum of all edge weights in the graph. The observed *proportion* of within-cluster edge strengths is then $\frac{1}{\beta}\sum_{ij} E_{ij}\delta(c(i), c(j))$, where the sum runs over all pairs of vertices, $c(i)$ is a membership function yielding the partition to which its vertex argument belongs, and $\delta(x, y) = 1$ if $x = y$ and 0 otherwise. Now define the *degree* of vertex i to be $\text{deg}(i) = \sum_j E_{ij}$, which tabulates the total strengths of all edges connecting i. A random graph *which respects the degree of each vertex* thus has an expected weight between vertices i and j of $\bar{E}_{ij} = \frac{1}{\beta}\text{deg}(i)\text{deg}(j)$ and, consequently, an expected *proportion* of within-cluster weights of $\frac{1}{\beta}\sum_{ij} \bar{E}_{ij}\delta(c(i), c(j))$. The modularity function characterizes the difference between the observed and expected proportions of intra-cluster strengths: $Q_{MOD}(C) = \frac{1}{\beta}\sum_{ij}(E_{ij} - \bar{E}_{ij})\delta(c(i), c(j))$. The Fast & Greedy algorithm is agglomerative, beginning with each vertex comprising its own partition. Partitions are repeatedly merged to produce the largest increase (or smallest decrease) in $Q_{MOD}(C)$, with the process repeating a total of $N - 1$ times to produce a final single partition. The resulting dendrogram is then cut at the height which yields the maximal value of Q, and the leaves of the cut define the optimal partitioning C^*.

Conversely, the Leading Eigenvector algorithm by Newman [25] attempts a divisive (or top-down) approach to maximizing the (same) modularity function $Q_{MOD}(C)$ by appealing to so-called *spectral* methods of traditional graph segmentation. To begin, all vertices are placed in the same partition and the goal is to allow modularity to guide the bisection of this partition into a new partitioning $C = \{C_1, C_2\}$. Note that $\delta(c(i), c(j)) = \frac{1}{2}(s_i s_j + 1)$ where s is an N-vector such that $s_i = 1$ if vertex $i \in C_1$ and $s_i = -1$ if $i \in C_2$. The modularity function can then be rewritten as $Q_{MOD}(C) = \frac{1}{2\beta}\sum_{ij}(E_{ij} - \bar{E}_{ij})s_i s_j$ (since $\sum_{ij} E_{ij} - \bar{E}_{ij} = 0$). In matrix form, defining $B = E - \bar{E}$ we have $Q_{MOD}(C) = \frac{1}{2\beta}s^T Bs$. B is known as the *modularity matrix*; the signs of each component of its 2nd (approximate) principal eigenvector indicate vertex membership in C_1 or C_2 (B thus takes the place of the graph Laplacian in traditional spectral graph segmentation methods; see Fortunato [22] for a more complete overview). The algorithm proceeds iteratively to compute a partitioning tree, splitting each node based on the signs of the 2nd principal eigenvector of the modularity matrix restricted to that node's vertex members. A branch terminates when its representative eigenvector has no differing signs; when all branches terminate the resulting dendrogram is searched recursively and cut at the height producing the maximal value of $Q_{MOD}(C)$.

The Walktrap algorithm (Pons and Latapy [26]) takes a completely different approach rooted in Markov chain theory. Assume a Markov chain with state space = C, and initially let $C = \mathcal{V}$ so that each vertex comprises its own partition. The transition matrix for this Markov chain is given by $P_{ij} = E_{ij}/\sum_k E_{ik}$ such that, at time t, the probability of transiting from $i \to j$ is P^t_{ij}. Note that we expect P^t_{ij} to be relatively large for strongly connected vertices. From this, a time-dependent distance is defined as $d^t_{ij} = \sqrt{\sum_{k=1}^{N}(P^t_{ik} - P^t_{jk})^2/deg(k)}$ where deg(k) is as above (but calculated from P^t instead of E). The number of steps, t is a required parameter. These distances d^t_{ij} are then input to Ward's algorithm [27] to choose two partitions to merge. Post-merging, the state space (and transition matrix P) are adjusted to reflect the new partition, repeating until a full dendrogram is produced which is again cut at the height producing maximal modularity. Unlike the Fast & Greedy and Leading Eigenvector methods, Walktrap does not use the modularity function to optimize the partitioning during tree building.

All above algorithms, as well as Infomap by Rosvall and Bergstrom [28] and Multilevel by Blondel et al. [29] perform relatively well for our SOM-based modifications (Section II-B) and are freely available in the `igraph` package [30]. Where applicable we use `igraph`'s default parameterizations (time steps $t = 4$ for Walktrap; number of trials = 10 for Infomap; the rest of the algorithms discussed here have no parameters).

B. Segmenting the SOM as a graph

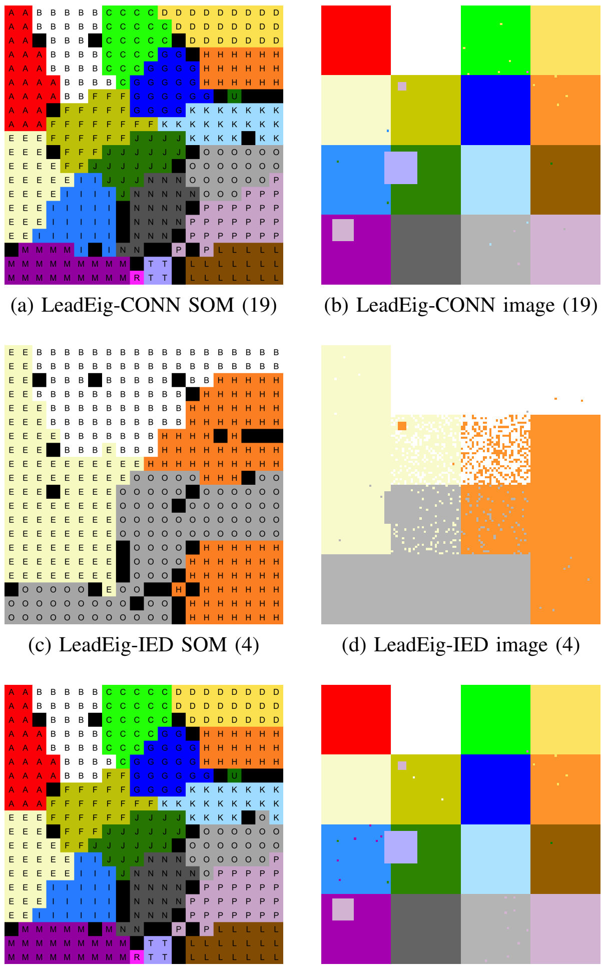
To utilize the graph segmentation paradigm for clustering each observation is represented as a vertex and edge weights between vertices are specified by pairwise point distances (usually Euclidean). For N observations, this requires storing and analyzing a graph with N vertices and $N(N - 1)/2$ edges which can be infeasible for many large, modern data sets. The relatively simple synthetic data cube described in III-A has 16,384 observations, requiring a distance structure of $\mathcal{O}(10^8)$ elements. We instead propose specifying a graph from learned SOM prototypes, typically requiring $\sqrt{N}$ vertices and $\sqrt{N}(\sqrt{N}-1)/2$ edges. Aside from the obvious computational and storage gains, prototype representations of the data (if learned properly) reduce noise inherent in large data sets, which should benefit algorithmic performance overall. Most notably, prototype-based graphs permit the specification of alternative similarity measures such as $CONN$. $CONN$ is itself a weighted, sparse adjacency matrix which naturally promotes its use in graph segmentation.

The following experiments highlight the drastic improvement that $CONN$ similarity can afford these classical graph segmentation algorithms. For comparison, we also experiment with inverse Euclidean distance (IED = $1/(1 + ED)$ where ED is the usual Euclidean distance) as a similarity measure. To isolate whether $CONN$'s weightings or its sparsity provide the greatest benefit, we further define a sparse IED (S-IED) similarity whose sparsity structure is forced to be that found in $CONN$. The prototype-based graphs are all processed in seconds (with $CONN$-weighted graphs requiring $<< 1$ second in most cases). We assess the quality of results by comparing to the known cluster structure using: 1) the adjusted Rand index (ARI) which gives the proportion of data point pairs assigned to the same cluster in both clusterings; 2) the Jensen-Shannon divergence between the distributions of cluster sizes; 3) visual inspection, particularly at regions where small cluster size or irregular properties (shape, size, density, proximities) prove difficult for clustering algorithms.

III. Data Analyses

A. Demonstration on a 6-D Synthetic Spectral Image

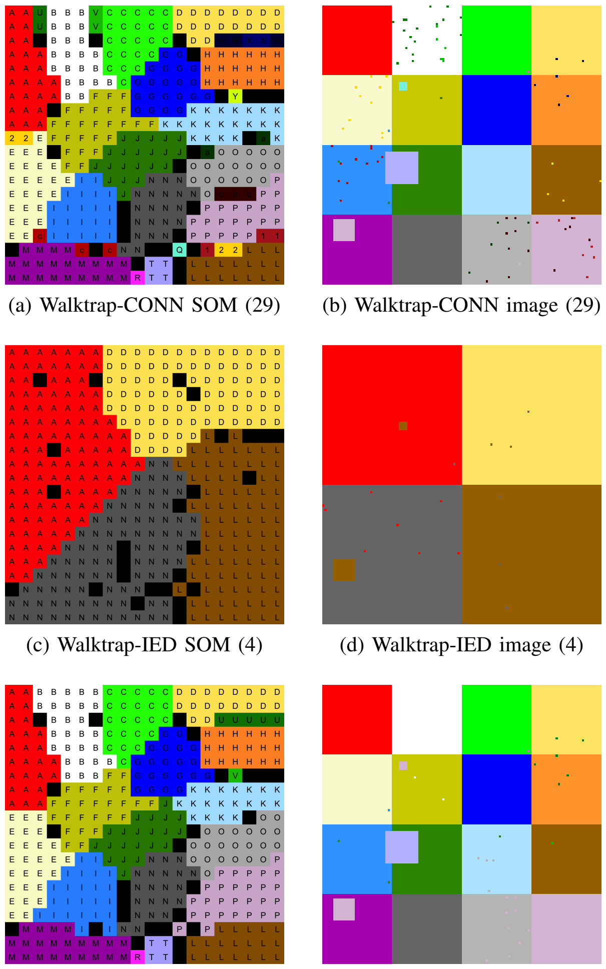
Our synthetic spectral image cube has 6-D feature vectors (the synthetic spectra) attached to each pixel location in a 128 x 128 pixel spatial area. This area is divided into 16 quadrants of 32 x 32 pixels, each quadrant representing a spectrally homogeneous region (a spectral class) as seen in Fig. 1 (d). In addition, four small classes are embedded in some quadrants: T (lilac), 16 x 16 = 256 pixels; S (light green), 8 x 8 = 64 pixels; Q (turquoise), 4 x 4 = 16 pixels; and R (magenta), a 1-pixel class (at the lower right corner of the green quadrant (class C), not visible at this resolution). Complete descriptions, including mean spectral signatures of the classes, is in Merényi et al. [9]. The feature vectors within each class were generated by adding $\approx 5\%$ Gaussian noise to all 6 dimensions of a representative class signature, so the resulting classes are spherical. However, the image has a non-trivial number of classes, the signatures of the different classes are highly similar in 6-D space (correlation coefficients of pairwise dimensions range from 0.0081 to 0.5641, Merényi et al. [9]), and it has rare classes, which are increasingly harder


(a) LeadEig-CONN SOM (19) (b) LeadEig-CONN image (19)

(c) LeadEig-IED SOM (4) (d) LeadEig-IED image (4)

(e) LeadEig-S-IED SOM (19) (f) LeadEig-S-IED image (19)

Fig. 2. Automated clusterings of SOM prototypes of the 6-D 20-class data cube by the Leading Eigenvector algorithm using CONN values (row 1), the inverse Euclidean distance (IED, row 2), and IED with CONN sparsity (row 3) between prototypes as similarity measure. Column 1 shows the segmented 20 x 20 SOM with letter labels (redundant with colors) also overlain, column 2 shows the spatial layout of the data clusters in image space. The numbers of extracted clusters are shown in parentheses.


(a) Walktrap-CONN SOM (29) (b) Walktrap-CONN image (29)

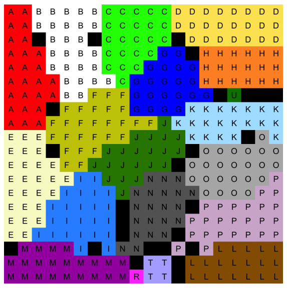
(c) Walktrap-IED SOM (4) (d) Walktrap-IED image (4)

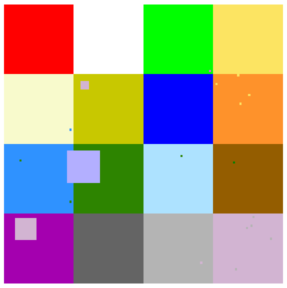
(e) Walktrap-S-IED SOM (20) (f) Walktrap-S-IED image (20)

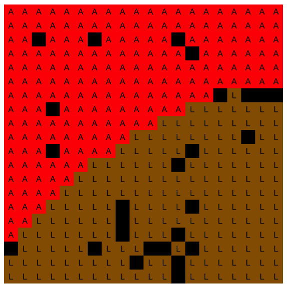
Fig. 3. Automated clusterings of SOM prototypes of the 6-D 20-class data cube by the Walktrap algorithm using CONN values (row 1), the inverse Euclidean distance (IED, row 2), and IED with CONN sparsity (row 3) between prototypes as similarity measure. Column 1 shows the segmented 20 x 20 SOM with letter labels (redundant with colors) also overlain, column 2 shows the spatial layout of the data clusters in image space. The numbers of extracted clusters are shown in parentheses.

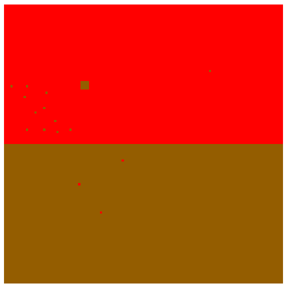
to find with decreasing size than the large classes. Interactive segmentation separated all clusters near-perfectly. This will be our reference clustering.

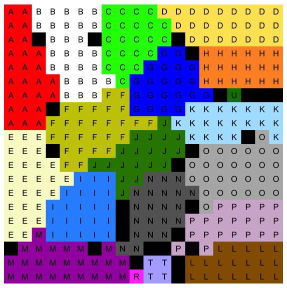
Figs 2–4 present the clustering results with the Leading Eigenvector (LE), Walktrap (WT), and Fast & Greedy (FG) algorithms, using the SOM prototypes as graph vertices and three different edge weightings described at the end of Section II-A. A summary is in Table I, which also includes results from the Infomap (IM) and Multi-Level (ML) algorithms. When these methods yield more than 20 clusters we add new labels / colors to indicate the extra clusters. First, all three methods perform generally well with CONN edge weighting as the E similarity matrix, and with inverse Euclidean distance when CONN sparsity is imposed on it (S-IED). In contrast, they all produce extremely poor results with inverse Euclidean distance (IED), finding only a few of the clusters, and some with considerable confusion (rather than superclusters). In these cases LE and FG show less salt-and-pepper type confusion than WT, but WT with CONN weighting is the only one that discovers the 16-pixel cluster Q (turquoise). All others confuse cluster Q with P (orchid), and S (light green) with O (gray) or P. However, all find the 1-pixel cluster R (magenta, as seen from the segmented SOM), and T (lilac). The explanation comes from Fig. 1 (a). Both R and T are completely isolated (pre-segmented) by CONN, while Q and S have connections to larger clusters that may make them candidates for merging in the judgement of the modularity objective function used by the graph segmentations. While LE and FG miss two of the small clusters by merging them with SOM-neighbor (thus similar) clusters, WT seems to have a tendency to oversegment and produce more salt-and-pepper type confusion. This may be explained by the bottom-up nature of WT combined with the fact that WT's merge decisions are not informed by modularity; WT only uses modularity to check where to cut the dendrogram once it has been built. In contrast, LE and FG use modularity to optimize the partitions. Overall, all methods perform very well with $CONN$ and S-IED measures.


(a) Fast&Greedy-CONN SOM (19)


(b) Fast&Greedy-CONN image (19)


(c) Fast&Greedy-IED SOM (2)


(d) Fast&Greedy-IED image (2)


(e) Fast&Greedy-S-IED SOM (19)

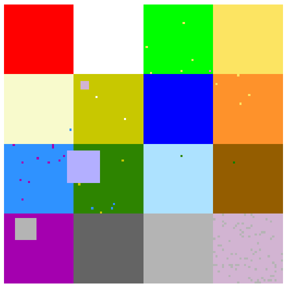

(f) Fast&Greedy-S-IED image (19)

Fig. 4. Automated clusterings from the SOM prototypes of the 6-D 20-class data cube by the Fast & Greedy algorithm using CONN values (row 1), the inverse Euclidean distance (IED, row 2), and IED with CONN sparsity (row 3) between prototypes as similarity measure. Column 1 shows the segmented 20 x 20 SOM with letter labels (redundant with colors) also overlain, column 2 shows the spatial layout of the data clusters in image space. The numbers of extracted clusters are shown in parentheses.

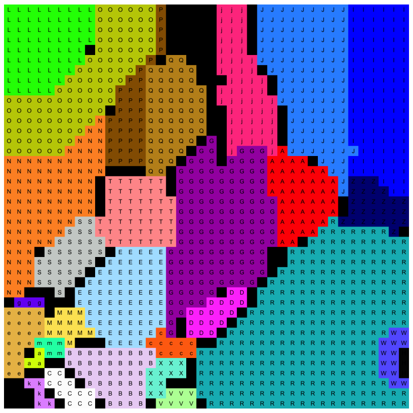
B. Results on Real Spectral Data Cube

We use bands 3–10 of a(n originally 12-band) spectral image of Ocean City, Maryland. The image comprises 512 x 512 pixels with 1.5 m/pixel ground resolution. Details of data acquisition are given in Csathó et al. [31], pre-processing and interactive segmentation to 28 verified land-cover classes are described in Merényi et al. [32]. While this image is only slightly higher-dimensional in feature space than the 6-D synthetic image, it has many clusters of variable statistical properties, shown in Table 1 of Merényi et al. [2], and it is very noisy. We compare our automatic segmentations with the interactive clustering from Merényi et al. [32], shown in Fig. 5 along with a partial list of classes. The $CONN$ representation of this data can be seen in Taşdemir and Merényi [10].

Cluster maps produced by the three top-performing methods are portrayed in Fig. 6, and discussed in the following. A summary of relative performances by all five algorithms (LE, WT, FG, IM and ML) is given in Table II.

TABLE I. Clustering evaluation of SOM-infused graph segmentation methods for the 6-D synthetic 20-class image compared to interactive SOM-clustering (20 clusters)

	graph segmentation method				
Similarity measure	Leading Eigenvector	Walktrap	Fast & Greedy	Infomap	Multi-Level
	Comparison by the number of clusters found				
CONN	19	29	19	32	19
IED	4	4	2	1	2
S-IED	19	**20**	19	25	19
	Comparison by Adjusted Rand Index				
CONN	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**
IED	0.29	0.32	0.12	0.0	0.12
S-IED	**0.99**	**0.99**	0.98	**0.99**	**0.99**
	Comparison by Jensen-Shannon Divergence				
CONN	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
IED	0.55	0.55	0.72	0.83	0.72
S-IED	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**


(a) Interactive-CONN SOM (28)

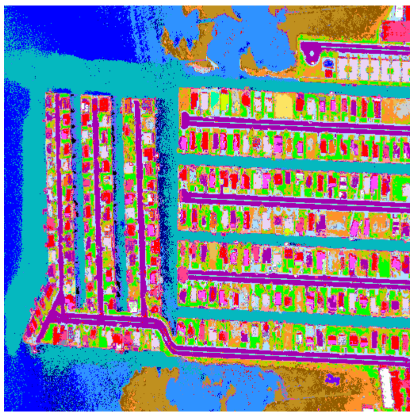
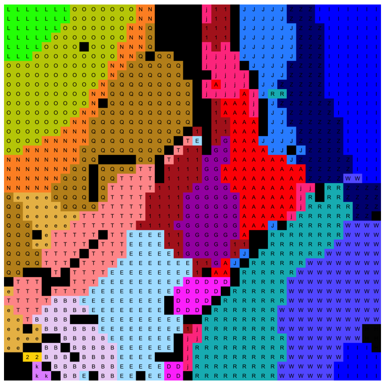
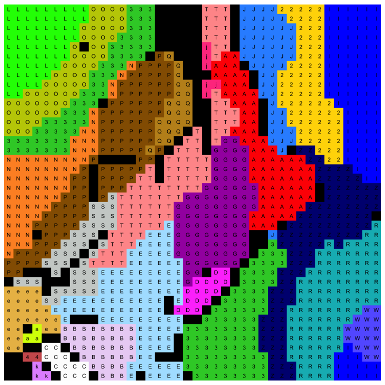
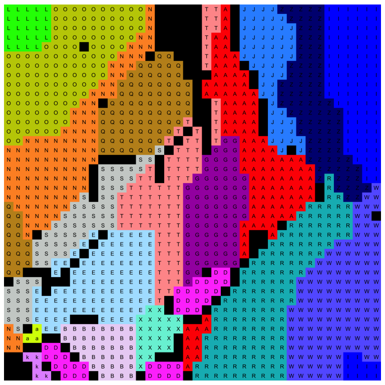
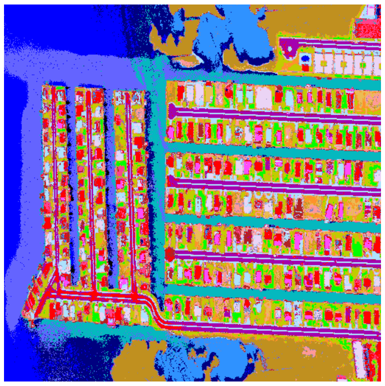

(b) Interactive-CONN image (28)

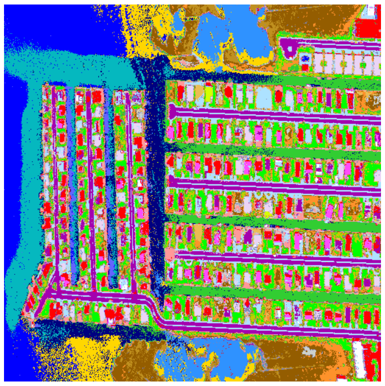
Fig. 5. Interactive clustering of the Ocean City data cube using CONNvis visualization, from Merényi et al. [32]. (a) The segmented 40 x 40 SOM with letter labels (redundant with colors) also overlain; (b) the spatial layout of 28 data clusters in the image space. This clustering has been verified by field knowledge. Clusters include ocean, bay, canal, pool water, (medium to dark blue colors); roofing materials of top of buildings (red, white, light pink, hot pink, magenta); grass, shrubs around houses (green, yellow), other vegetation (orange, brown), and several rare clusters (roofs a, m, V, shrub g). Asphalt (magenta, G) and reflective paint (light blue, E) occur on both roads and roofs. The number in parentheses indicates the number of extracted clusters.

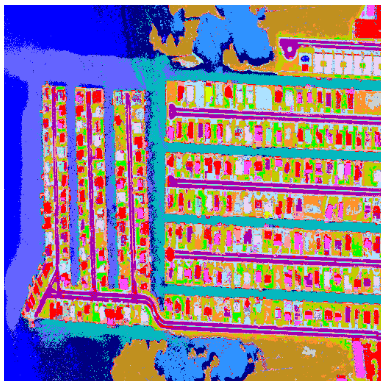

(a) LeadEig-CONN SOM (21)

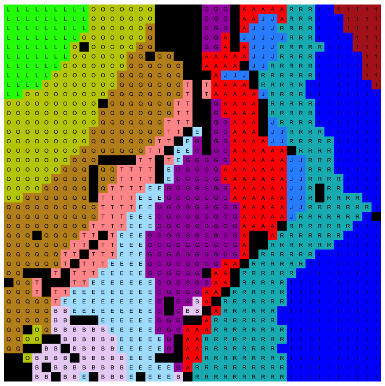

(b) Walktrap-CONN SOM (25)

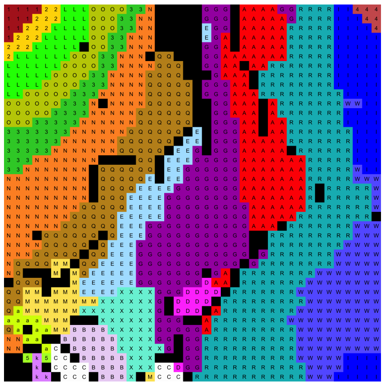

(c) Fast&Greedy-CONN SOM (19)

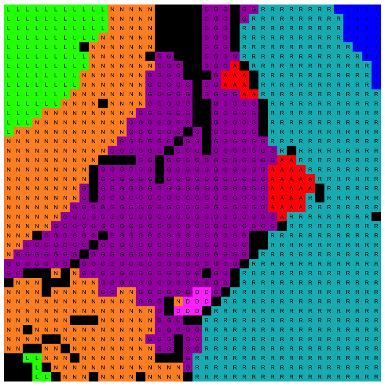

(d) LeadEig-CONN image (21)

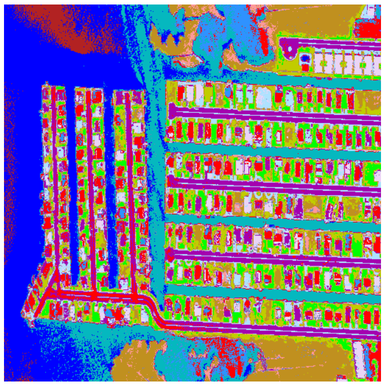

(e) Walktrap-CONN image (25)

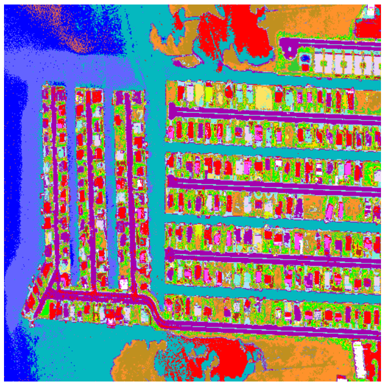

(f) Fast&Greedy-CONN image (19)


(g) LeadEig-S-IED SOM (12)


(h) Walktrap-S-IED SOM (22)


(i) Fast&Greedy-S-IED SOM (7)


(j) LeadEig-S-IED image (12)


(k) Walktrap-S-IED image (22)

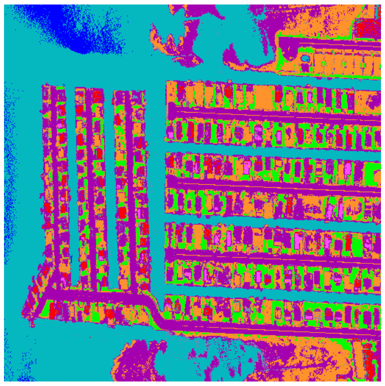

(l) Fast&Greedy-S-IED image(7)

Fig. 6. Automated clusterings of SOM prototypes of the Ocean City data cube by the Leading Eigenvector (column 1), Walktrap (column 2), and Fast & Greedy (column 3) algorithms; using $CONN$ (rows 1–2) and S-IEDP (rows 3–4) similarity measures. Rows 1 and 3 show the respective segmented 40 x 40 SOMs with letter labels (redundant with colors) also overlain, rows 2 and 4 show the spatial layout of the data clusters in the image space. Numbers in (parentheses) indicate the number of clusters identified by each method.

TABLE II. CLUSTERING EVALUATION OF SOM-INFUSED GRAPH SEGMENTATION METHODS FOR THE OCEAN CITY IMAGE CUBE COMPARED TO INTERACTIVE SOM-CLUSTERING (28 CLUSTERS)

	graph segmentation method				
Similarity measure	Leading Eigenvector	Walktrap	Fast & Greedy	Infomap	Multi-Level
	Comparison by the number of clusters found				
CONN	21	**25**	19	84	20
IED	2	3	2	1	2
S-IED	12	22	7	41	10
	Comparison by Adjusted Rand Index				
CONN	0.51	0.52	**0.58**	0.23	**0.58**
IED	0.16	0.30	0.16	0.0	0.16
S-IED	0.44	0.46	0.35	0.34	0.39
	Comparison by Jensen-Shannon Divergence				
CONN	0.04	0.03	0.05	0.32	0.04
IED	0.47	0.40	0.47	0.62	0.47
S-IED	0.08	**0.02**	0.26	0.12	0.13

Similarly as with the 6-D 20-class synthetic image cube, all algorithms do very poorly when using the IED similarity measure, identifying 2–3 superclusters of the 28 known clusters, *i.e.,* generally no salt-and-pepper confusion, but no useful detail is returned. This approach does not seem to be able to utilize the discriminating power of the spectral information. We omit the IED cases from figures and concentrate on the performance with the $CONN$ and the S-IED (IED with $CONN$ sparsity) measures. Among these, LE, FG and WT produce better clusterings than IM or ML when considering the quality indicators in Table II as well as the cluster localization by visual inspection (not shown for IM and ML). For space considerations we only include LE, WT and FG in Fig. 6.

First, on this real data set, the clusterings using $CONN$ measure are markedly better than those using S-IED, for all methods. This is indicated by both the smaller number of clusters found and the poorer scores by the ARI and JSD measures, for the latter, in Table II. (The only exception is the JSD score of WT by a very small margin.) Given this, we further narrow our discussion to the three $CONN$-weighted segmentations (in rows 1–2 of Fig. 6). With the $CONN$ measure all three methods did remarkably well. Visual inspection reveals further characteristics of the performances. While the general structure of the segmented SOM by FG looks most similar to the interactively segmented SOM in Fig. 5 (a), it returns the lowest number of clusters (19). WT comes closest to the confirmed number (28), with 25 clusters, but this is mostly the result of splitting some of the large water and vegetation classes (causing some unfavorable visual impression). WT, however, is the only one that detects the difference of the two brown clusters (P, Q). These are spectrally very similar but with a consistent, real difference. LE seems to merge the largest number of relevant clusters and, at the same time, split other (both large and small) clusters. The merges by all three methods are most often with SOM-neighbor clusters, thus reasonable in many cases (despite the loss of some details like, for example, finer distinction among roofing materials). WT and FG both discover the rare cluster a, but none delineate m, or V, which are roofing materials of rare occurrence in the image. Cluster S (dry dirt / grass patches) is completely overtaken by cluster T in the LE map.

Despite their differences, all methods localize the major structures of ocean and canal water, bays, shrubs, lawn; and most manmade objects like houses, roads, center lane of road, drive-ways, very well. Many of the differences in cluster delineation occur in areas of the SOM where $CONN$vis (not shown here) indicates thinning of the manifold over wide swaths, therefore the resulting confusions are not entirely unreasonable. The 1.5 m / pixel ground resolution naturally mixes material classes in pixels where object boundaries fall (for example where grass is adjacent to a house). The small scale on which different materials alternate in this image generates a large number of such mixed pixels, which cause unavoidable confusion. Given this and the high level of noise, the relatively low ARI scores (around 50%) can be considered quite good for this image. Visual impression confirms this.

In summary, these graph segmentation methods, with the use of SOM prototypes and $CONN$ similarity measure, show great potential for high-quality automation of SOM segmentation in the case of large data cubes with complex structure.

IV. Discussion, Conclusion and Outlook

Common to the five algorithms we evaluate is that they do best when their input is the $CONN$ graph (SOM prototypes with $CONN$ similarity as edge weighting). Informing the IED adjacency matrix with CONN sparsity (S-IED) dramatically improves the outcome of the IED-based methods, but they still significantly underperform the $CONN$ edge weighting. Considering that (under mild conditions, and assuming correct SOM learning) the $CONN$ graph is the weighted Delaunay graph of the n-D data space (Martinetz and Schulten [21], Taşdemir and Merényi [10]) where the weighting senses the "weak seams" in the manifold in both global and local relations, this is perhaps not surprising.

We conclude that, SOM prototypes alone as input to graph segmentation algorithms — while reducing computation time and storage demands by magnitudes — do not help produce clusterings with a quality anywhere near that of interactive cluster extraction, for complicated large data sets. In contrast, employing the inherently sparse $CONN$ weighted adjacency matrix produces results that approach the detail and quality of interactive SOM clustering, and the SOM segmentation takes negligible time, $\ll 1$ second. A further, distinct benefit is that the automated approach can utilize more nuanced information from the $CONN$ matrix than the human operator, which can result in more complete labeling of the SOM.

In this paper we cluster somewhat complex synthetic data, and complex, noisy, but relatively low-D real data, whose cluster structures are known or have been verified in previous studies. Therefore, formal evaluation against known "templates" can be made. These data sets facilitate experimentation with a wider set of graph-segmentation algorithms for systematic and more thorough isolation of the relative advantages than we can do while using very complex data for the purpose of new discoveries. In a recent study (Merényi et al. [11]) we show automatic segmentation of a 200-D hyperspectral radio astronomy image from ALMA (Atacama Large Millimetre Array) which identifies intricate physical structure in a protoplanetary disk. In that case, we show that the SOM-powered graph-segmentation discovers structural features interpretable for astronomers but not detected by more traditional methods. There, we do not have a "template" (beyond our interactive clustering) to use in formal evaluation, for lack of previous works that produced comparable clusterings. However, we can still say that, for ALMA data — with much higher feature dimension, many clusters but considerably less noise — WT works best followed by IM. This cautions that, depending on general data characteristics, which can significantly vary for different types of Big Data (*e.g.,* terrestrial hyperspectral imagery, astronomical imagery, fMRI data cubes) different combinations of segmentation algorithm and similarity measure may be best. In future work we will attempt systematic consideration of suitable trade-offs.

Finally, we want to note that SOM learning itself may be a lengthy iterative process. To alleviate that bottleneck parallel implementations can be used (see, *e.g.,* Lachmair et al. [33]) to bring the overall processing time of cluster discovery to a level that scales with Big Data.

References

[1] T. Kohonen, *Self-Organization and Associative Memory*. New York: Springer-Verlag, 1988.

[2] E. Merényi, K. Taşdemir, and L. Zhang, "Learning highly structured manifolds: Harnessing the power of SOMs," in *Similarity-Based Clustering*, ser. Lecture Notes in Computer Science, M. Biehl, B. Hammer, M. Verleysen, and T. Villmann, Eds. Berlin Heidelberg: Springer Verlag, 2009, vol. 5400, pp. 138–168.

[3] T. Villmann, R. Der, M. Herrmann, and T. Martinetz, "Topology Preservation in Self–Organizing Feature Maps: Exact Definition and Measurement," *IEEE Transactions on Neural Networks*, vol. 8, no. 2, pp. 256–266, 1997.

[4] E. Bodt, M. Cottrell, and M. Verleysen, "Statistical tools to assess the reliability of self-organizing maps," *Neural Networks*, vol. 15, pp. 967–978, 2002.

[5] L. Zhang and E. Merényi, "Weighted Differential Topographic Function: A Refinement of the Topographic Function," in *Proc. 14th European Symposium on Artificial Neural Networks (ESANN'2006)*. Brussels, Belgium: D facto publications, 2006, pp. 13–18.

[6] A. Ultsch, "Clustering with SOM: U*c," in *Proc. 5th Workshop on Self-Organizing Maps (WSOM 2005)*, Paris, France, September 5–8 2005, pp. 75–82.

[7] L. Hamel and C. Brown, "Improved interpretability of the unified distance matrix with connected components," in *Proc. 7th International Conference on Data Mining (DMIN'11)*, CSREA Press, Las Vegas, 18-21 July 2011, pp. 338–343.

[8] M. Cottrell and E. de Bodt, "A Kohonen map representation to avoid misleading interpretations," in *Proc. 4th European Symposium on Artificial Neural Networks (ESANN'96), D-Facto*, Bruges, Belgium, April 1996, pp. 103–110.

[9] E. Merényi, A. Jain, and T. Villmann, "Explicit magnification control of self-organizing maps for "forbidden" data," *IEEE Trans. on Neural Networks*, vol. 18, no. 3, pp. 786–797, May 2007.

[10] K. Taşdemir and E. Merényi, "Exploiting data topology in visualization and clustering of Self-Organizing Maps," *IEEE Trans. on Neural Networks*, vol. 20, no. 4, pp. 549–562, 2009.

[11] E. Merényi, J. Taylor, and A. Isella, "Mining complex hyperspectral ALMA cubes for structure with neural machine learning," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec 2016, pp. 1–9.

[12] J. Vesanto and E. Alhoniemi, "Clustering of the self-organizing map," *IEEE Trans. Neural Networks*, vol. 11, no. 3, pp. 586–600, 2000.

[13] M. Cottrell and P. Rousset, "The Kohonen algorithm: A powerful tool for analyzing and representing multidimensional quantitative and qualitative data," in *Proc. Int. Work-Conf. Artificial Neural Networks*, 1997, pp. 861–871.

[14] F. Murtagh, "Interpreting the Kohonen self-organizing feature map using contiguity-constrained clustering," *Pattern Recognition Letters*, vol. 16, no. 4, pp. 399–408, 1995.

[15] D. Brugger, M. Bogdan, and W. Rosenstiel, "Automatic cluster detection in kohonen's som," *IEEE Transactions on Neural Networks*, vol. 19, no. 3, pp. 442–459, 2008.

[16] M. Goncalves, M. Netto, and J. A. F. Costa, "A new method for unsupervised classification of remotely sensed images using Kohonen Self-Organizing Maps and agglomeration hierarchical clustering methods," *Intl. Journal of remote Sensing*, vol. 11, no. 29, pp. 3171–3207, 2008.

[17] K. Taşdemir, P. Milenov, and B. Tapsall, "Topology-based hierarchical clustering of self-organizing maps," *IEEE Transactions on Neural Networks*, vol. 22, no. 3, pp. 474–485, 2011.

[18] W. Liao, H. Chen, Q. Yang, and X. Lei, "Analysis of fMRI data using improved self-organizing mapping and spatio-temporal metric hierarchical clustering," *IEEE Transactions on Medical Imaging*, vol. 27, no. 10, pp. 1472–1482, 2008.

[19] K. Taşdemir, "Spectral Clustering as an Automated SOM Segmentation Tool," in *Proc. Workshop on Self organizing Maps (WSOM 2011)*, 2011, pp. 71–78.

[20] K. Taşdemir and E. Merényi, "A validity index for prototype based clustering of data sets with complex structures," *IEEE Trans. Systems, Man and Cybernetics, Part B*, vol. 41, no. 4, pp. 1039–1053, August 2011, dOI: 10.1109/TSMCB.2010.2104319.

[21] T. Martinetz and K. Schulten, "Topology representing networks," *Neural Networks*, vol. 7, no. 3, pp. 507–522, 1994.

[22] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3-5, pp. 75 – 174, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/B6TVP-4XPYXF1-1/2/99061fac6435db4343b2374d26e64ac1

[23] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E*, vol. 70, p. 066111, Dec 2004. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.70.066111

[24] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Phys. Rev. E*, vol. 69, p. 066133, Jun 2004. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.69.066133

[25] ——, "Finding community structure in networks using the eigenvectors of matrices," *Phys. Rev. E*, vol. 74, p. 036104, Sep 2006. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.74.036104

[26] P. Pons and M. Latapy, "Computing communities in large networks using random walks," in *Proceedings of the 20th International Conference on Computer and Information Sciences*, ser. ISCIS'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 284–293. [Online]. Available: http://dx.doi.org/10.1007/11569596_31

[27] J. H. Ward, "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/01621459.1963.10500845

[28] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proceedings of the National Academy of Sciences*, vol. 105, no. 4, pp. 1118–1123, 2008. [Online]. Available: http://www.pnas.org/content/105/4/1118.abstract

[29] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008. [Online]. Available: http://stacks.iop.org/1742-5468/2008/i=10/a=P10008

[30] G. Csardi and T. Nepusz, "The igraph software package for complex network research," *InterJournal*, vol. Complex Systems, p. 1695, 2006. [Online]. Available: http://igraph.org

[31] B. Csathó, W. Krabill, J. Lucas, and T. Schenk, "A multisensor data set of an urban and coastal scene," in *Int'l Archives of Photogrammetry and Remote Sensing*, vol. 32, no. 3/2, 1998, pp. 26–31.

[32] E. Merényi, B. Csató, and K. Taşdemir, "Knowledge discovery in urban environments from fused multi-dimensional imagery," in *Proc. IEEE GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas (URBAN 2007).*, P. Gamba and M. Crawford, Eds., Paris, France, 11–13 April 2007, pp. 1–13.

[33] J. Lachmair, E. Merényi, M. Porrmann, and U. Rückert, "A reconfigurable neuroprocessor for self-organizing feature maps," *Neurocomputing*, vol. 112, pp. 189–199, 2013.