Neural Computing and Applications (2020) 32:18161–18178
https://doi.org/10.1007/s00521-019-04198-6

WSOM 2017

Empowering graph segmentation methods with SOMs and CONN similarity for clustering large and complex data

Erzsébet Merényi[1,2] · Joshua Taylor[1]

Received: 2 May 2018 / Accepted: 11 April 2019 / Published online: 21 June 2019

Abstract
High-dimensional, large, and noisy data with complex structure challenge the limits of many clustering algorithms including modern graph segmentation methods. SOM-based clustering has been shown capable of capturing many clusters of widely varying statistical properties in such data. However, to date the best discovery results are produced by interactive extraction of clusters from informative SOM visualizations. This does not scale for Big Data, large archives, or near-real-time analyses. We approach this challenge by infusing SOM knowledge into leading automatic graph segmentation algorithms, which produce extremely poor results when segmenting the SOM prototypes without this information, and which would take a prohibitively long time to segment the input data sets. The knowledge translation occurs by casting the SOM prototypes as vertices and the CONN similarity measure as edge weightings of a graph which is then presented to graph segmentation algorithms. The resulting performance closely approximates the precision of the interactive SOM segmentation for complicated data and, at the same time, is extremely fast and memory-efficient. We demonstrate the effectiveness on a simple synthetic data set and on a very realistic fully labeled synthetic hyperspectral image. We also examine performance dependence on available parametrizations of the graph segmentation algorithms, in combination with parametrizations of the CONN similarity measure.

Keywords SOM clustering · Graph segmentation · CONN similarity · Big Data · Automation

1 SOM clustering for complex data

Self-Organizing Maps (SOMs [13]) 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 [20]). 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 $i, i = 1, \ldots, N$, each with a prototype (weight vector) $\mathbf{w}_i \in \mathscr{R}^n$ attached to it, learns the structure of a given data manifold $M \subset \mathscr{R}^n$ comprising S samples $\{\mathbf{x}_l\}_{l=1}^{S}$, $\mathbf{x}_l = (x_{l1}, \ldots, x_{ln}) \in M$, typically $S \gg N$. 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 including topology preservation, the prototypes provide faithful approximation of the data distribution and their ordering on the lattice reflects their similarity relationships in data space. For this paper, we assume that the correctness of learning has been verified. (See some overview of verification methods and references in [20, 21].) Further information that can be derived from a learned SOM includes the number of data points mapped to each prototype (visualized as a "hit map"), the data space distances of lattice neighbor prototypes (U-matrix and variants, [10, 34], octagonal erosion [4], mU-matrix, [18]), or more involved quantities like connectivity of prototypes (CONN) by [31]. These can help locate contiguous groups

✉ Erzsébet Merényi
erzsebet@rice.edu

Joshua Taylor
jtay@rice.edu

1 Department of Statistics, Rice University, Houston, TX 77005, USA

2 Department of Electrical and Computer Engineering, Rice University, Houston, TX 77005, USA

of similar prototypes in the SOM grid where each group collectively represents 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 [20]. It is in such cases where interactive segmentation tends to produce better quality results than automated methods, but this requires expertise and can be time-consuming. Consequently, this does not scale with the demands of near-real-time processing, autonomous situations, or large archives, where it is most needed.

1.1 Objectives of this work

We present an automated approach to SOM segmentation which closely approximates the details obtained with interactive segmentation for complicated data and which is very fast and memory-efficient. We achieve this by infusing SOM knowledge into cutting-edge graph segmentation algorithms which, by themselves, produce extremely poor results when given the customary pairwise distances of the SOM prototypes. We show that using an SOM-derived similarity measure as edge weightings for graph-cutting algorithms brings a break-through in their performance. This is in agreement with previous experiments on several different real data sets of considerable complexity, where domain experts' judgment and comparison against independent results on the same data served as quality assessment [21–23]. Here, we provide formal evaluation against labeled "templates" for a highly structured extremely realistic synthetic hyperspectral image in addition to evaluation on simple synthetic data. In previous work we utilized default parameterizations of the graph segmentation methods; in this paper we also explore available parametrizations to assess the effect on clustering quality.

1.2 Previous work in automation of SOM segmentation

Several automated approaches segment 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 [35], Ward measure [5] or centroid linkage constrained by grid neighborhood contiguity [24] have been demonstrated to work well in HAC for relatively low-D data with a small number (3–8) of clusters. In [5], more than 30 clusters are captured from 12-D time series data, but no evaluation of how these clusters match relevant known groupings is given. Another proposition [2] is to generate a smooth function called the Clusot surface via mixtures of "modified" Gaussians where the standard deviations of the Gaussians are direction-dependent and computed from the (normalized) data space distances to neighbor prototypes in the respective directions in the lattice, weighted by the winning frequencies. The resulting surface has valleys where large prototype distances coincide with low winning frequencies, which is then flooded to detect peaks 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 by [9] restricting the merging to lattice neighbor prototypes and excluding, from the HAC phase, dead and heterogeneous prototypes from transition regions between clusters. The latter are identified by high relative dispersion of the pixel features in their receptive field. This results in capturing four to five clusters of land cover from IKONOS and Landsat5 3-band imagery with very high (92–99%) accuracy.

More complex imagery is segmented by [33]. Four million 20-D spectral signatures are learned with a 50×50 SOM, which is then clustered with HAC methods to compare the effects of Ward, centroid, average, and CONN linkages. The HAC with CONN linkage, which is derived from the CONN similarity [31], generally outperforms the others, as well as *k*-means. Since our automated approach also relies on this measure, we review CONN briefly below. HAC is applied to SOMs of fMRI data using a novel spatiotemporal distance measure composed of pairwise correlations of prototypes weighted by an exponentially decaying function of their lattice distance in [15]. While the correlation measure admits varied cluster shapes, the combined measure may fail at sharp boundaries where the manifold is discontinuous. The relatively large weighting generated by close lattice proximity of two prototypes across such a boundary may counteract the small correlation of those prototypes in data space. This could explain why their clustering success is limited to 3–4 brain areas.

As an alternative to HAC [30] 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. 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 2-D data sets have specific challenges such as variable cluster shapes and densities versus proximities, whereas other data may have clusters more balanced in size and other properties.

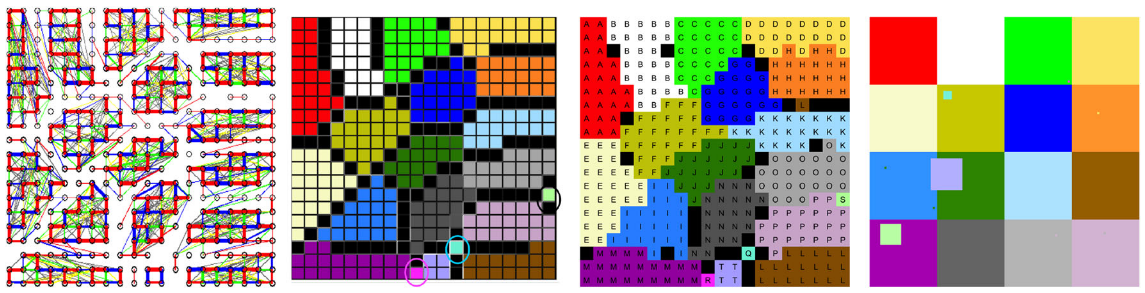
Previous works illuminate that while clustering difficulties can be caused by volume, feature dimensionality, the number of inherent partitions and noise, the complexity of a manifold—and thus the clustering challenge—ultimately depends on the variations in cluster sizes, shapes, densities, and the number and relative positions of clusters. We aim to address the combination of these challenges.

1.3 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 [31, 32]. The connection strength CONN(i, j) between prototypes attached to neurons i and j is measured as the number of data vectors which choose one prototype as their first, and the other as second SOM winner. The visualization of the CONN matrix (CONNvis) over the SOM can 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 Sect. 2.4. (In spectral images, n-D feature vectors are attached to (x, y) spatial locations; these are the input vectors to SOM learning.) In Fig. 1a, the SOM lattice of 20 $\times$ 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 gray 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. As proven in [16] (under mild conditions), two prototypes get connected if they are non-empty Voronoi neighbors in data space, and they are chosen by (at least one) data points as a pair of first and second SOM winners. 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 (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. 1a that clusters have many relatively weak violating intra-cluster connections which do not interfere with cluster identification, and connections across clusters are weak or missing. CONN informs about the mapping quality, both visually (with CONNvis overlay) and through quantitative CONN-based topology measures [20].

Cluster boundaries are found between regions that are strongly connected inside and have thin or no connections to other regions. For visualization, a nonlinear binning of CONN strengths is applied to aid the human eye. The bin boundaries are automatically derived from CONN statistics using the means of the kth-ranking connections of all prototypes ($k = 1, \ldots, N$). Details on this and a cluster



(a) CONNvis of SOM **(b)** Interactive-CONN SOM **(c)** Plurality labels on SOM **(d)** Interactive-CONN image

Fig. 1 **a** Visualization of CONN values (CONNvis) over the 20 $\times$ 20 SOM of the 6-D 20-class data cube described in Sect. 2.4. 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** Plurality labels of samples mapped to each prototype. Letter labels (redundant with colors) are also indicated. Differences with Fig. 1b occur because a few samples map to prototypes in other classes (due to noise or imperfect 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 classes except for a few stray pixels extraction procedure are given in [31]. This binning also provides a sensible automatic thresholding scheme for removing unimportant connections below the kth mean. In this work, to keep experiments to a mangeable number, we elect to use only one nonzero threshold for CONN values, determined in a data-dependent fashion as the mean of the fourth-ranking connections ($k = 4$). The fourth-ranking mean has been our empirically selected default in CONN visualizations as it has worked well for the human analyst. By using the same CONN threshold for the `igraph` methods, we are assessing whether the support given to the human expert benefits the automated algorithms. (Exploration of value thresholds based on considerations of further CONN statistics is the subject of a follow-up study.) Connections can also be filtered by length. As elaborated in [31] connections of length larger than one can be classified as globally (severely) or locally topology violating. Locally violating connections are between a prototype $\mathbf{w}_i$ and those of its Voronoi neighbors that are *connected* in data space and fit within the tightest possible SOM neighborhood around $\mathbf{w}_i$ with length > 1. The connected Voronoi neighbors comprise the *masked* Delaunay graph proven by [16] to be built by SOM learning and represent the topology of the data. Within the lattice radius of the above tightest neighborhood, we can consider a connection non-violating and thus harmless for cluster identification. For the 6-D 20-class synthetic data cube, the maximum number of connected Voronoi neighbors is 16, which can fit within a lattice radius of two [31]. Globally violating connections in a well-learned SOM are typically caused by a few noisy samples, and therefore, their removal can be beneficial for segmentation. (This assumes that the SOM is free of strong globally violating connections.) Thresholding by length can also be informed by the mean strength of the connections of given length and by the percentage of data samples involved in those connections. All of this information is recorded in the connection statistics.

Figure 1b shows the SOM lattice with clusters of similar prototypes extracted interactively from the CONNvis representation. In Fig. 1c, each prototype is labeled by the (known) plurality class of the samples mapped to it. Slight differences with Fig. 1b are caused by a few samples mapping to prototypes not in their classes (due to noise, or imperfect SOM learning). Notably, 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. 1d which shows the clusters in data space. This clustering is almost a perfect match to the true classes.

2 Graph-based clustering of SOMs

All clustering schemes require a representation of the data to be clustered and specification of some measure describing the (dis-)similarity among members of this representation. A graph $\mathscr{G}$ is a convenient tool to both organize and represent this information: The collection of N vertices $\mathscr{V}$ represents the N members to be clustered and an $N \times N$ adjacency matrix $\{A_{ij}\}$ encodes the relationship (defined by the chosen (dis-)similarity) between vertices i and j. A is often binary, but we will work with an extension of this which permits an adjacency matrix E whose (i, j) entries represent a *graded* similarity between i and j. A partitioning $C = \{C_1, \ldots, C_{nC}\}$ of $\mathscr{V}$ into nC mutually exclusive sets defines clusters of vertices. Using the information contained in E (or A), graph-based clustering schemes optimize (over C) a measure $Q(C)$ representing some pre-defined *quality* of the partitioning C.

Modularity is a popular, graph-specific choice for $Q(C)$ which measures intra-cluster strengths *relative* to those that would be expected from a random partitioning, subject to certain constraints. Given a partitioning of vertices $C = \{C_1, \ldots, C_{nC}\}$, let $\beta = ||\mathrm{vec}(E)||_1$, 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 $\delta(a, b) = 1$ if $a = b$ and 0 otherwise. Now define the *degree* of vertex i to be $\deg(i) = \sum_j E_{ij}$, 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 $\bar{E}_{ij} = \frac{1}{\beta}\deg(i)\deg(j)$ and 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: $Q_{\mathrm{MOD}}(C) = \frac{1}{\beta}\sum_{ij}(E_{ij} - \bar{E}_{ij})\delta(c(i), c(j))$. Many of the graph clustering methods discussed below utilize the concept of modularity in some way.

2.1 Background on algorithm classes

The Fast & Greedy algorithm [3] is an extension of the modularity-based algorithm of [25], fine-tuned for computational performance. The agglomerative approach begins with each vertex comprising its own partition. Two partitions are repeatedly merged to produce the largest increase (or smallest decrease) in $Q_{\mathrm{MOD}}(C)$, with the process repeating $N - 1$ times to produce a final single partition. The resulting dendrogram is then cut to yield the maximal value of $Q(C)$, and the leaves of the cut define the optimal partitioning C^*. Because Fast & Greedy builds an entire clustering tree it requires no operating parameters. The Multilevel algorithm [1] is similar to Fast & Greedy, but (unlike Fast & Greedy) if no gain is possible for $Q_{\text{MOD}}(C)$ no merge occurs. After all vertices have been examined a new graph is formed whose vertices represent the partitions found in stage one; both intra- and inter-partition weights are summed from each partition's constituent vertices. This process repeats in a hierarchical fashion until no further increase in modularity is possible via merging. As an exhaustive agglomerative algorithm, there are no parameters governing Multilevel's performance, but because the initial vertex examined is randomly chosen, repeated applications of the method are suggested.

Conversely, the Leading Eigenvector algorithm [26] attempts a divisive (or top-down) approach to maximizing the (same) modularity function $Q_{\text{MOD}}(C)$ 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 $Q_{\text{MOD}}(C) = \frac{1}{2\beta}\sum_{ij}(E_{ij} - \bar{E}_{ij})s_i s_j$ (since $\sum_{ij} E_{ij} - \bar{E}_{ij} = 0$). In matrix form, defining $B = E - \bar{E}$ we have $Q_{\text{MOD}}(C) = \frac{1}{2\beta}s^T Bs$. B is known as the *modularity matrix*; the signs of each component of its second (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 [8] for a more complete overview). The algorithm proceeds iteratively to compute a partitioning tree, splitting each node based on the signs of the second principal eigenvector of the modularity matrix restricted to that node's vertex members. A branch terminates when either its representative eigenvector has no differing signs, or (optionally) its length reaches a pre-defined value; when all branches terminate, the resulting dendrogram is cut at the height producing the maximal value of $Q_{\text{MOD}}(C)$. Since branch length is an optional parameter we have left it unspecified in the following results, allowing the tree's growth to be completely governed by the spectral modularity approximation.

The Walktrap algorithm [27] takes a completely different approach rooted in Markov chain theory to form its candidate partitions. 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 $P_{ij} = E_{ij}/\sum_m E_{im} = E_{ij}/\deg(i)$ such that, at time t, the probability of the transition $i \rightarrow j$ is P^t_{ij}. Note that we expect P^t_{ij} to be relatively large for strongly connected vertices unless their degree is made large by many weak edges. From P_{ij}, a time-dependent distance is defined as $d^t_{ij} = \sqrt{\sum_{m=1}^N (P^t_{im} - P^t_{jm})^2/\deg(m)}$ where $\deg(m)$ is as above. These distances d^t_{ij} are then input to Ward's algorithm [36] to choose two partitions to merge. Post-merging, the state space (and transition matrix P) is 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. The number of steps t is a required parameter.

Departing completely from modularity, the Infomap algorithm [28] determines clusters based on a cost function $Q(C)$ guided by an information-theoretic analysis of random walks on graphs. Utilizing the same transition probabilities P as in Walktrap, $Q(C)$ is constructed to describe the total entropy of movement both between and within clusters: $Q(C) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{r=1}^{nC} p^r_{\circlearrowright} H(\mathcal{P}^r)$, where $q_{\curvearrowright}$ is the probability of moving between clusters and $p^r_{\circlearrowright}$ is the probability of movement within cluster r, $H(\mathcal{Q})$ is the entropy of between-cluster movement and $H(\mathcal{P}^r)$ is the entropy of movement within cluster r. These quantities can all be derived directly from the entries of P. Since minimum entropy corresponds to the most information about a stochastic system, the goal is thus to *minimize* Q with respect to C. Because direct minimization of Q is in most cases computationally intractable, a greedy agglomerative partitioning scheme is devised, much like Fast & Greedy, except merging occurs to minimize $Q(C)$ in this case. Owing to random initiation of neighbor evaluation, the algorithm likely produces a local minimizer of Q. To account for this, the entire process is repeated *num.trials* times (which is the only parameter required) and the best global minimizer of Q is selected.

When graphs are infused with SOM-specific knowledge (outlined below), all of the above methods produce recognizably meaningful clusterings of our test data sets; the main difference among them is the level of detail they uncover in more complex cases. We have utilized their implementations available in the `igraph` package [6] available for the R programming language.

2.2 SOM-specific graph knowledge

Traditional use of the graph segmentation paradigm for clustering involves representing each observation as a vertex, with E_{ij} a function of pairwise point distances (usually Euclidean). For S observations, this requires storing and analyzing a graph with S vertices and $S(S -$

$1)/2$ edges which can be infeasible for many large, modern data sets. The relatively simple synthetic data cube described in Sect. 2.4 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 $\sqrt{S}$ vertices and $\sqrt{S}(\sqrt{S}-1)/2$ edges. Contrary to large observation-level graphs, the prototype-based graphs are all feasible to store and are processed in seconds (with CONN-weighted graphs requiring $\ll 1$ s in most cases).

Our experiments show the drastic improvement that CONN similarity affords graph segmentation algorithms. Prototype-specific measures such as CONN harness information about the manifold (such as density and topology) which traditional distance-based similarities lack. To highlight CONN's capabilities, we will employ inverse Euclidean distance ($\mathrm{IED} = 1/(1+\mathrm{ED})$ where ED is the usual Euclidean distance) as a benchmark similarity measure. CONN is, additionally, extremely sparse when compared to IED; to isolate whether its sparsity structure or its actual values are the largest contributor to its success we also experiment with a sparse-IED (S-IED) graph whose edge weights are taken from IED but whose sparsity inherits that of the CONN (that is, edge weights equal IED where CONN is nonzero, and equal 0 elsewhere). This modification dramatically improves IED-based performance.

2.3 Evaluation methods

For comparative purposes, all `igraph` methods were supplied with graphs derived from each of the CONN, IED and S-IED similarity measures. For CONN and S-IED, all combinations of the CONN value thresholdings tv $\in$ {0, mean of fourth-ranking connections} and length thresholdings tl $\in$ {2, 3 or 4} were considered. These are discussed in Sect. 1.3. The expected effect of thresholding is (a) severing unimportant edges; (b) increasing the probability P_{ij} (Sect. 2.1) due to lowering the degree $\deg(i)$ of a node i while leaving the E_{ij}, remaining after thresholding, unchanged. This can increase the modularity of relevant subcommunities or the chance of visiting those (by Walktrap). For IED, which makes no use of CONN values, no threshold applies. Thus each method produced $2 \times 6 + 1 = 13$ clusterings. We ran Walktrap with 2, 4, 6, 8, and 10 steps; accordingly, the list of methods compared here consists of the interactive clustering performed by the human analyst (I-C), Infomap (abbreviated IM), Fast & Greedy (FG), Leading Eigenvector (LE), Multilevel (ML), and Walktrap with steps 2, 4, 6, 8, 10 (WT(2, 4, 6, 8, 10)). WT(4), which is Walktrap's default parameterization in `igraph`, produced the most promising clustering in previous work [23] therefore one interest is to see if non-default parameters may improve its relative performance. LE's parameterization involves specifying the number of eigenvectors to compute (instead of the default maximum number); we do not explore this because, theoretically, allowing less than the maximum number of eigenvectors cannot produce better results. Similarly, we use IM's default parameter of 10 repeated runs because fewer runs are unlikely to produce better results in general. While we may explore increasing the number of runs in the future, we first study other parameters that are likely to have more impact. By all comparisons FG, LE, and ML performed equally on the data sets studied here, therefore we will represent this group under the collective label "ML" and only display ML's results for space considerations. FG, LE and ML all attempt direct modularity maximization, so their self-similar performance is unsurprising.

We assess the quality of all results by comparing them to known reference cluster structure using several measures. For quantitative assessment, we compare the number of clusters returned from each clustering as well as two cluster-matching indices: the Adjusted Rand Index (ARI, [11]) which reports the corrected-for-chance proportion of data point pairs assigned to the same cluster in both clusterings, and the Normalized Mutual Information measure (NMI, [7]) which compares the information content in the reference and candidate clusterings. For qualitative assessment, we perform visual comparison to reference images. Particular scrutiny is given to regions where small cluster size or irregular properties (shape, size, density, proximities) prove difficult for clustering algorithms.

To facilitate visual inspection and discussion, all candidate clusterings have undergone a process of *reconciliation* to the reference clustering which involves assigning known reference cluster labels to the best matching cluster returned from the automated methods. The matching is done by assigning, for example, reference label A to the candidate cluster which has the highest Jaccard Similarity Coefficient (JSC) with reference cluster A. The JSC is calculated at the observation level (not at the SOM prototype level, even though the candidate clustering is performed there) to better incorporate the characteristics of the cluster in feature space. We stress that in this process only the label is changed; candidate clusters remain otherwise the same (in terms of their observation-level membership, size, shape, etc.).

2.4 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 128×128 pixel spatial area. This area is divided into 16 quadrants of 32×32 pixels, each quadrant representing a spectrally homogeneous region (a spectral class) as seen in Fig. 1d. In addition, four small classes are embedded in some quadrants: T (lilac), 16 × 16 = 256 pixels; S (light mint green), 8 × 8 = 64 pixels; Q (turquoise), 4 × 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, are in [18]. The feature vectors within each class were generated by adding ≈ 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, highly similar signatures in 6-D space (correlation coefficients of pairwise dimensions range from 0.0081 to 0.5641 [18]), and it has rare classes, which are increasingly harder to find with decreasing size. Interactive segmentation separated all clusters near-perfectly.

Figure 2 shows representative clusterings of the 20-class synthetic cube as returned by ML and WT(4) for CONN thresholding combinations. The value threshold, tv, was chosen as the mean of the fourth-ranking connections (tv = 5.82 for this data set) while the length threshold of tl = 3 was chosen because the length of non-globally violating connections is ≤ 2 for this data set, and tl = 2 yields the same visual results. See Sect. 1.3 for principles of value and length thresholding of CONN. For compactness, we present only CONN results since the S-IED similarities performed nearly identically for these data. IED performs poorly (generating only 1–3 superclusters as demonstrated in [21]) and we exclude it from reporting. We note for these data all methods (except for IED-based ones) perform comparably, with the exception of IM which oversegments (as reported in [21]) and suffers as a result. The number of clusters (*nC*) in each clustering, as well as the agreement measures (ARI, NMI) relative to the truth image, is reported under each panel of Fig. 2.

For the synthetic 20-class data, all methods produce clustered images that are recognizably similar to the truth. Value thresholding results in a noticeable salt-and-pepper effect of the resulting images for both methods (comparing the right panels of Fig. 2a and d to g and j for ML, and Fig. 2b and e to h and k for WT(4)). This is due to the more than doubling of the number of determined clusters in this case, which is itself a result of the increased sparsity that value thresholding at the 5.82 level introduces into CONN. While the increased confusion from value thresholding may seem detrimental, it also allows ML to distinguish the small turquoise cluster (Q). Thresholding CONN at length 3 with tv = 0 (no value thresholding) also produces mixed effects: The mint green cluster (S) is separated from the orchid colored cluster (P) but not as cleanly as possible, grouping multiple non-S pixels together to form S. We point out, however, that the over-segmentation occurs at cluster boundaries that involve almost-empty prototypes with weak connections; the true, larger clusters—which are strongly connected within themselves—have not been split.

In general, increasing either the CONN value or length thresholds beyond a certain point would ultimately result in over-segmentation of the image; this process becomes first noticeable at cluster boundaries, as can be verified by comparing the SOMs of Fig. 2a, b to their counterparts in, particularly, Fig. 2g–k. While the human brain would easily classify the bottom two rows of Fig. 2 as inferior, agreement measures (ARI, NMI) on the whole do not suffer as the cluster boundaries begin their dissolution due to thresholding. Because of this, while using numerical summary statistics such as these to find an "optimal" thresholding value might seem attractive, we urge caution in using them alone as their information summarization is not necessarily sensitive to what a human analyst would call a good clustering.

2.5 Results on complex, high-D data: the Megascene hyperspectral image

We use a hyperspectral urban image, part of the "Megascene" synthetically generated at the Rochester Institute of Technology through the DIRSIG modeling procedure [12, 29]. This image is nearly indistinguishable from a real hyperspectral image and it comes with truth labels for all pixels. It comprises 400 × 400 pixels in 210 image bands in the 0.38 to 2.4 μm visible-near-infrared spectral window, with a spatial resolution of 2 m/pixel. The image looks very realistic as seen in Fig. 3a. It contains 72 different surface materials, including seven tree species and grasses, about two dozen roof covers, a similar number of sidings, paving and building materials (bricks of different brands and colors, stained woods, vinyl, painted metals), and car paints. Preprocessing consisted of removal of (simulated) atmospheric effects and bad bands, conversion of radiances to reflectances, and brightness normalization to remove linear illumination effects [17, 19]. The multitude of clusters, with widely varying statistical properties in 184-D feature space (after exclusion of bad bands), presents a formidable clustering task. Clusters extracted from a 40 × 40 SOM interactively are shown in the spatial image, and in the SOM, in Figs. 3 and 4, respectively. The correctness of the clusters can be checked against a map of truth labels (Fig. 3c). In this truth image, we mask out the three largest classes (K, T, V, medium green, salmon, and light green colors, respectively, in Fig. 3b). This makes the variety of the smaller classes visible. We will also use this masked truth map, as well as the masked interactive clustering (I-C) in Fig. 3d, for the evaluation of clusterings, as explained below. Figures comparing the spectral characteristics of a

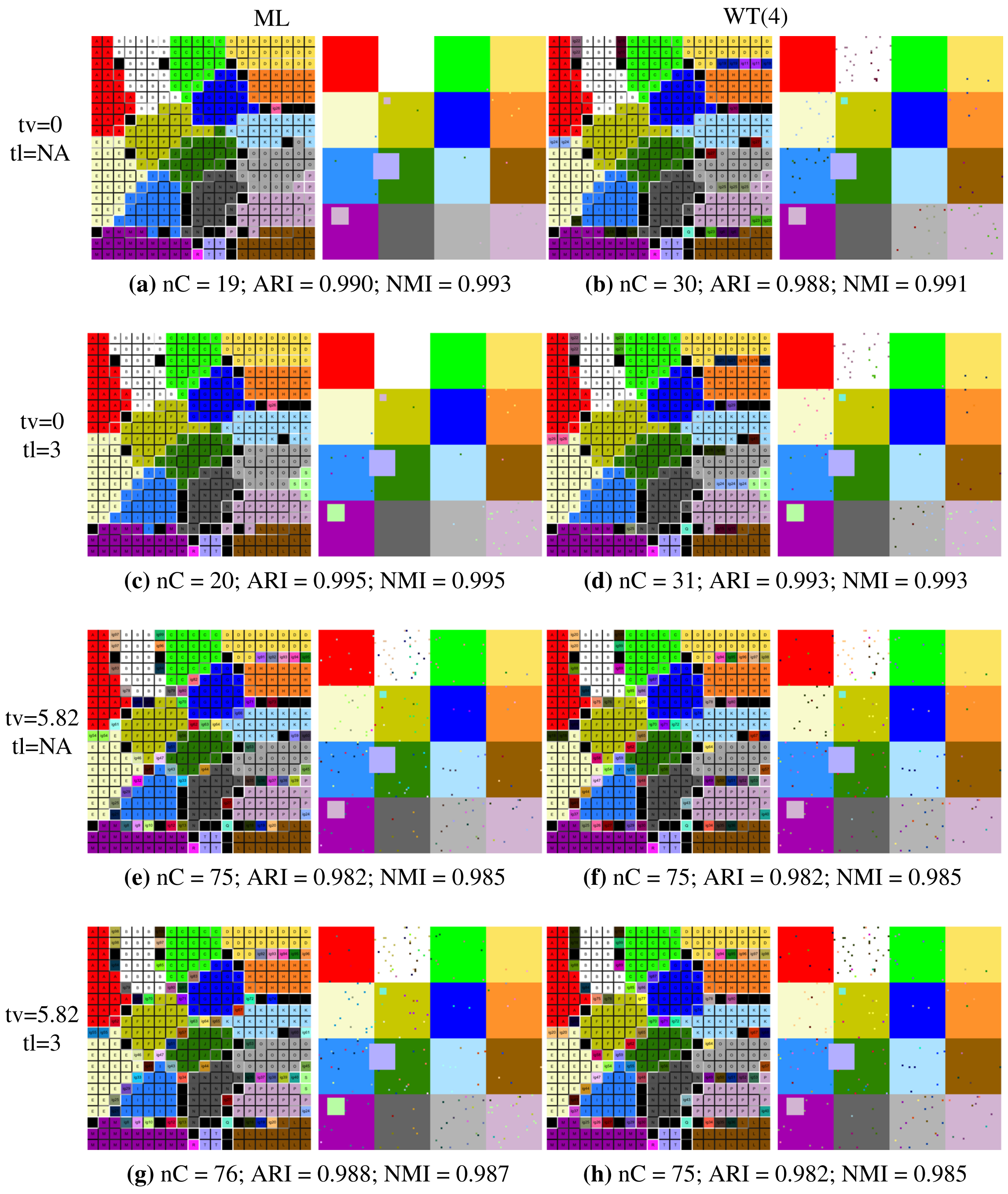


Fig. 2 Automated clusterings from the SOM prototypes of the 6-D 20-class data by ML (column 1) and WT(4) (column 2) algorithms using CONN similarity measure thresholded at: tv = 0, tl = NA (row 1); tv = 0, tl = 3 (row 2); tv = 5.82, tl = NA (row 3); tv = 5.82, tl = 3 (row 4). For each panel, the segmented 20 × 20 SOM is presented on the left and the resulting clustered image is on the right. The number of clusters returned from each algorithm (nC), along with the ARI and NMI agreement measures, is reported under each panel

number of the I-C clusters with true classes are available in [19] and show excellent match.

All igraph experiments were conducted and evaluated as described in Sect. 2.3. For CONN and S-IED, CONN value and connection length thresholdings {tv = 0, tv = 20.79}, {tl = NA, tl = 3, tl = 4} were used. tv = 20.79 is the mean of rank 4 connections. ML continues to be the collective label for IM, FG, LE, ML which performed equally on this data set too. The clusterings by igraph methods are compared, separately, to both the truth image (Fig. 3c) and to the I-C (Fig. 3d). From all comparisons, we exclude the large classes K, T, V (as in Figs. 3 and 4), for two reasons: (a) The SOM of these data is very fragmented within these large classes, which

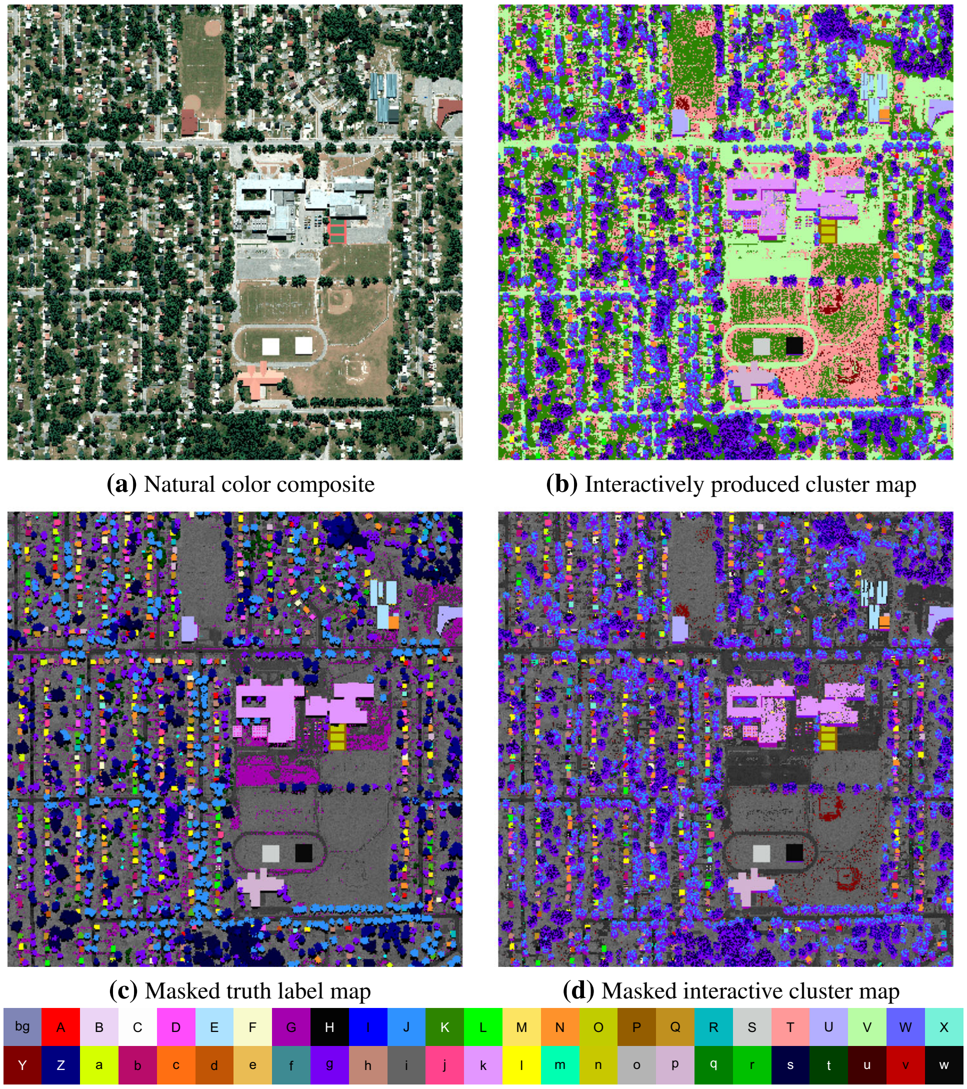

(a) Natural color composite
(b) Interactively produced cluster map
(c) Masked truth label map
(d) Masked interactive cluster map
bg A B C D E F G H I J K L M N O P Q R S T U V W X
Y Z a b c d e f g h i j k l m n o p q r s t u v w

Fig. 3 **a** A natural color composite combined from red, green, and blue spectral bands of the 400 × 400 pixel, 210-band synthetic hyperspectral Megascene image [12, 29]. It covers 800 × 800 square meters (2 m/pixel). Surface cover types in this image include trees, grass, pavings, major buildings, tennis court, obvious in the image; a variety of roof and other building materials (most on residential homes, the material variety not obvious from this natural color image); several types of car paint (each manifesting in 2–3-pixel areas, not visible here). Altogether 72 classes of extremely varied sizes and spectral properties are present. **b** Interactively obtained cluster map (I-C) from the SOM in Fig 4, left. Material labels are coded by color and letter as keyed at the bottom. **c** The map of truth labels, overlain a grayscale image of the scene, with the large background clusters K, T, V (grass, paved roads, and lots) masked (removed) to provide better contrast for the many different roof materials, and other unique material types such as tennis courts. Twenty of these clusters are roof types of different spectral characteristics, some with subtle discriminating features. **d** The same, I-C cluster map as in **b**, with the K, V, and T classes removed

puts the `igraph` algorithms at a disadvantage and no meaningful comparison could be made for those areas. (The human analyst has no difficulty identifying those large classes despite this fragmentation, thanks to the mU-matrix "fences" and overall connectivity structure.) Addressing this situation is deferred to future work. (b) These large classes would dominate comparison statistics and obscure the performance of the more important and more challenging smaller classes; and also visually suppress them.

Comparisons are done by three quality indicators: (1) Adjusted Rand Index (ARI). Normalized Mutual Information (NMI) gives the very same trends, and therefore, we omit it for space considerations; (2) Visual observation; (3) The number of clusters coinciding with clusters in the reference map. By each of these indicators, two relevant subsets of the clusters are evaluated separately: The group we call "Vegetation Clusters" and the group we call "Small Clusters". Vegetation Clusters (g, G, J, Z, s, t) comprise the two larger maroon (G) and purple (g) clusters

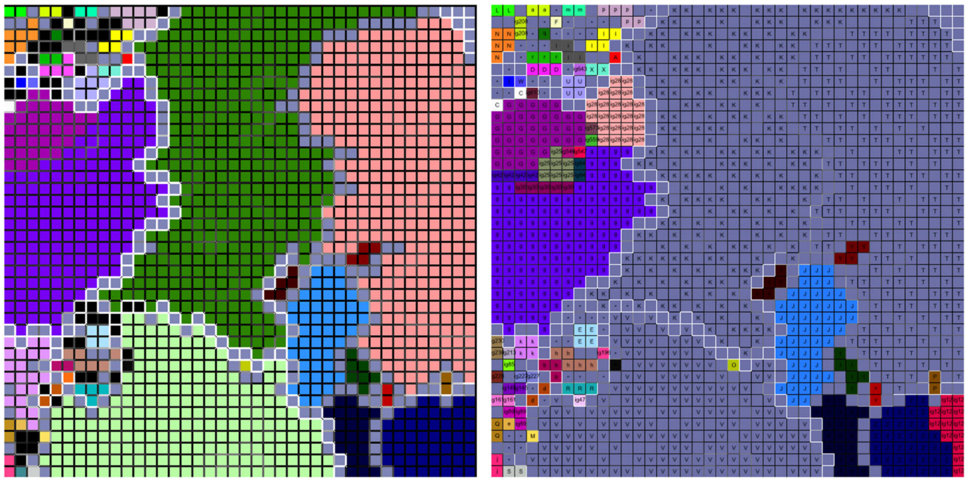


Fig. 4 Left: the SOM that produced the cluster map in Fig. 3b by interactive segmentation. The discovered clusters are color-coded as keyed in Fig. 3. The cells with medium gray color, appearing mostly along major cluster boundaries, are "dead" SOM prototypes with no data points mapped to them. Some cells (shown as black) were left unclustered by the human expert, for reasons of colors limitations, and partly because of uncertainty. These are among the groups of Small Clusters in the upper left and lower left. The white "fences" (an alternative visualization, mU matrix, described in [18]) indicate major cluster boundaries. Weaker boundaries are suppressed in this view. Right: the same SOM as at left, as clustered by the top-performing automatic method, WT(4). The large classes K, T, V are masked out to show the areas excluded from evaluations. The letter labels of the clusters are also overlain for convenience. The masked out prototypes as well as the dead prototypes all have the gray "bg" color. This is also an example of comparison of clusterings on the SOM prototype level, which we do (but not show) for every clustering. The coincidence of delineated clusters between the I-C and WT(4) clustering is remarkable

at the center left of the SOM in Fig. 4, left, and the medium blue, dark green, and dark blue clusters at the lower right. The Small Clusters are the rest of the clusters seen in the SOM (after exclusion of the K, T, V clusters). The rationale for separate assessment of these groups is that the members of the Small Clusters in this image are too small compared to the Vegetation Clusters. They range in size from 1 pixel to a few hundred pixels (less than 1% of the image) each, some less than 1/1000 of the image pixels. Statistics for the union of the two groups would not reveal the quality of the Small Clusters. Accurate identification of small-sized clusters—such as those identifying different roofing materials—is important from the point of view of a clustering challenge, and for a user.

Comparison of the igraph clusterings to both the truth label map (Fig. 3c) and the I-C (Fig. 3d), as reference images, is done for the following reasons: (a) The I-C discovered a few clusters which are not labeled as unique in the truth map, but which are justified based on their spectral characteristics. The igraph clusterings tend to identify the same "new" clusters. One example is the grass in the two baseball diamonds, outlined in rust color (label Y) in the open field in the lower right of Fig. 3d while the truth label map (Fig. 3c) indicates no difference there. These new—legitimate—clusters lower the clustering quality scores when comparing with the truth map. Comparison to the I-C provides a way to reward their discoveries. The igraph methods may also discover more clusters where the human analyst left SOM areas unclustered. (b) The truth label map has some inconsistencies. The most conspicuous example is the labeling of some pixels as tree species while the spectral signatures in those pixels strongly differ from vegetation spectra. This can happen when substrates are seen through trees; or strong shadows are cast on part of the trees. The large cluster g (purple) collects such signatures and consequently mixes with (makes up about half of) the actual tree species (very dark blue, purple, and lighter blue in the truth map) in a salt-and-pepper fashion. Figure 5 illustrates the mislabeling. This is also an example of how the labeling on the object level may not always correspond to the spectral distinctions on the pixel level; therefore, delineating the actual feature (spectral) variations is important. Figure 6 presents the distributions of CONN versus IED similarity values, which we discuss below to give insight for some observations.

A summary comparison of methods with the truth image and with the I-C by the Adjusted Rand Index (ARI) is displayed in Figs. 7 and 8, respectively. The bar charts show the relative agreements with the respective reference maps for all methods in three parts, one each for the three similarity measures (CONN, S-IED, and IED) used by graph segmentation methods. Within each part, the indices are shown for the Vegetation Clusters and the Small Clusters groups separately. Over each bar, the number of

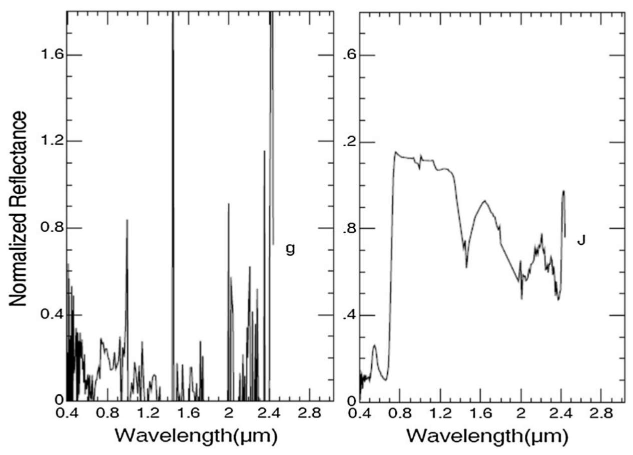


Fig. 5 Left: the spectral signature of pixels in cluster g. Right: the signature of cluster J, Norway Maple tree (and that of other trees) is dramatically different from the cluster g spectrum

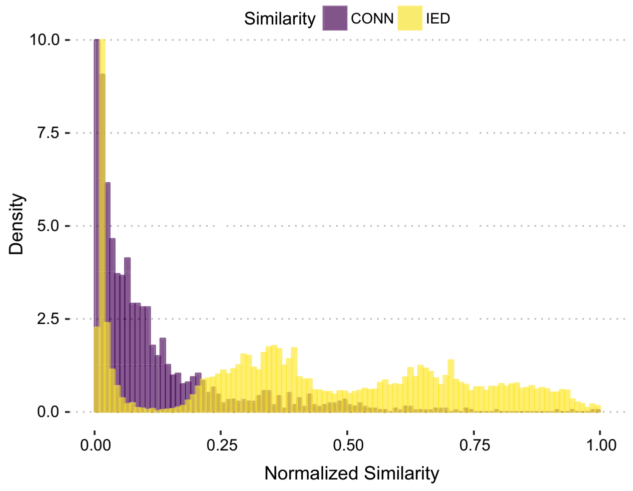


Fig. 6 Distribution of CONN and IED similarity values for the 40 × 40 SOM of the Megascene data cube

discovered clusters that coincide with clusters in the reference map is indicated.

Visual judgment of the relative fitness of all clusterings was made by inspecting the respective segmentations of the SOM (only WT(4) is shown, in Fig. 4, for space considerations); and the cluster maps, taking into account the matches within both the group of Vegetation Clusters (g, G, J, Z, s, t) and the group of Small Clusters. This assessment considers the spatial distribution of mismatches on the pixel level. Given two clusters each with 50% match to a reference map, one with matching pixels concentrated in a coherent area and mismatched pixels at cluster boundaries; and another with scattered mismatching pixels, the visual score is much worse for the latter. The statistical indices would assign the same score to both.

The visual ranking of the Megascene cluster maps agrees with the ARI ranking with minor differences. Cluster maps produced by two of the top-performing automated methods and parameter combinations, and reconciled to the I-C and the truth image, are compared in Figs. 9 and 10, respectively. The results are discussed below along with ARI evaluations.

First (as with the 6-D 20-class image) all methods do very poorly with IED similarity, identifying 2–3 superclusters, i.e., no confusion but no useful detail. This is evident from the ARI bar charts and from cluster maps, and also consistent with experiments on other data in previous work (see images in [21, 23]). (The ARI value is made high in this case by the large superclusters, not by good match of many relevant clusters.) The histograms of the CONN versus IED values (Fig. 6) offer insight: CONN values (purple) are sparse at the high-similarity end, while this measure judges the majority of the data point pairs as dissimilar, providing good discrimination among clusters. In contrast, IED values are almost evenly spread over the high-similarity range, with only a small fraction of the values falling into the low-similarity end, suggesting much less discriminating power than CONN. We omit the IED cases from figures and further discussion and concentrate on the performance of methods with the CONN and the S-IED (IED with CONN sparsity) measures. Among these, WT(4) or WT(2) produce better clusterings than IM, ML, and WT(6–10) based on visual assessment. This is further elaborated through the formal ARI scores.

Second, methods using the CONN measure generally score a little higher than using S-IED, both in comparison to the I-C and the truth map, with a few exceptions. This difference is more noticeable visually (both in the SOMs and in the cluster maps) than in the ARI charts.

The third general observation is that, *by visual inspection*, all methods/similarity measure combinations do better with CONN value thresholding tv = 20.79. The ARI charts contradict in a few cases, which we point out below. Experiments with connection length thresholding tl = 3 and tl = 4 produce negligible visual difference or, if at all, slightly favor tl = 3. We select tl = 3 based on CONN statistics: The length of non-globally violating connections is 2 for the Megascene, and tl = 3 leaves those connections intact. Connections longer than 3 involve less then 0.5% of the data, and have a mean connection strength lower than the tv = 20.79 threshold.

As measured by ARI, the fitness of each automated method, chosen similarity measure and thresholding scheme is very similar when compared to the truth map (Fig. 7). The Vegetation group is large but the image contains only 7 known tree species. (Grasses, T and V, are excluded from this study.) Yet CONN and S-IED-based methods yield many clusters in this group (anywhere from thirty to sixty depending on method/similarity combination). The I-C produced six Vegetation Clusters, much

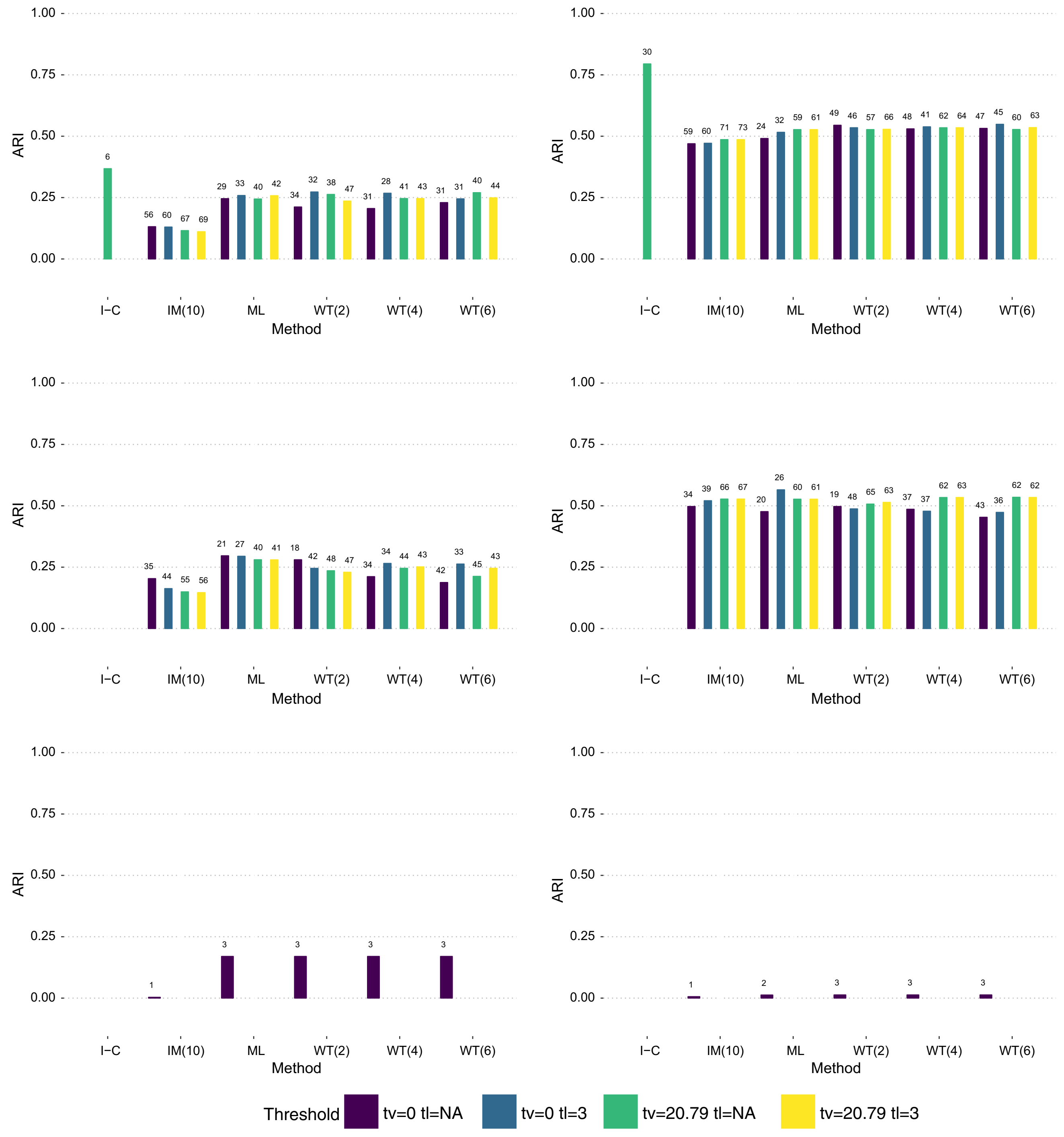


Fig. 7 Adjusted Rand Index values for comparing clusterings by `igraph` methods with the truth labels (Fig. 3c). The left and right columns show, separately, the indices for the "Vegetation" and the "Small" classes, respectively. The number over each bar indicates how many clusters the given method/parameter combination produced for the respective subset. I-C: Interactive clustering; ML: Multilevel (represents FG, LE, and ML, which perform equally); WT(x): Walktrap with x allowed steps. The colors signify CONN thresholding as described in Sect. 2.2. tv $= val$: removing connections with strength (CONN value) below val; tl $= length$: removing connections longer than *length*. Indices shown for methods using the CONN (top row), the S-IED (center row), and the IED similarity measure (bottom row)

more in line with the number and homogeneity of the known vegetation classes. Consequently, the I-C achieves the highest (but still relatively low) ARI score. IM severely splinters the Vegetation Clusters, and its ARI score suffers accordingly. Comparison to the Small Clusters in the truth map tells mostly the same story, with all methods

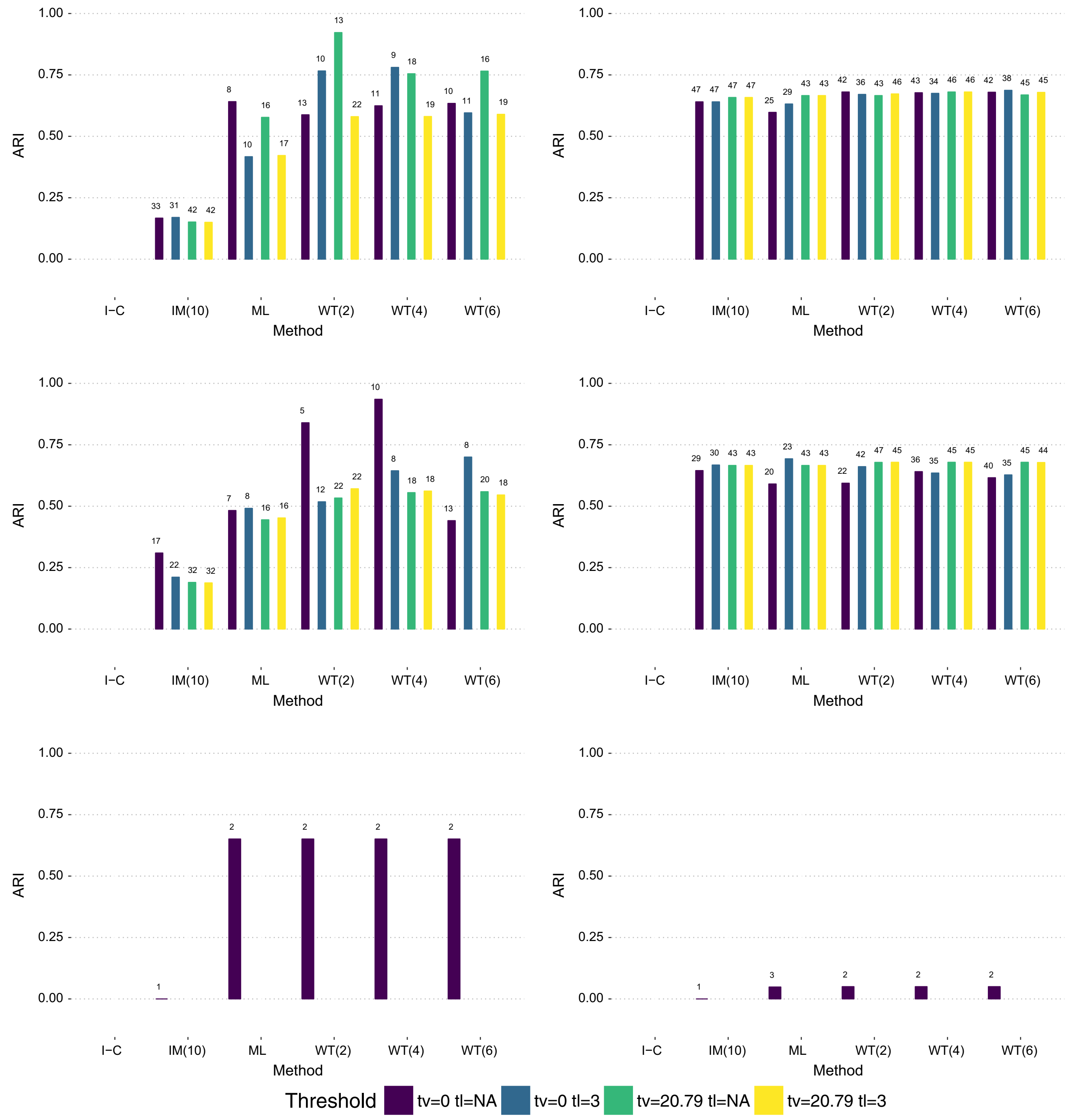


Fig. 8 Adjusted Rand Index for comparing clusterings by `igraph` methods with the I-C (Fig. 3d). The left and right columns show, separately, the indices for the "Vegetation" and the "Small" classes, respectively. The number over each bar indicates how many clusters the given method and parameter combination produced for the respective subset. Acronyms and notations are as in Fig. 7. Indices shown for methods using (top row) CONN, (center row) S-IED, and (bottom row) IED similarity measure

performing even more similarly (and overall with higher ARI scores around 0.50). The I-C is far superior in isolating meaningful small details with an ARI $\approx$ 50% higher in this group than the rest. We note, however, that the confusion between the largest "vegetation" cluster, g, and the truth labels for trees (as shown in Fig. 5) is largely responsible for the generally low ARI score of the Vegetation Clusters.

Compared to the I-C (Fig. 3d), the ARI performance of all methods (Fig. 8) is again about equal for the Small Clusters but with an overall improvement of scores hovering around 0.70. Inspecting the Vegetation Clusters

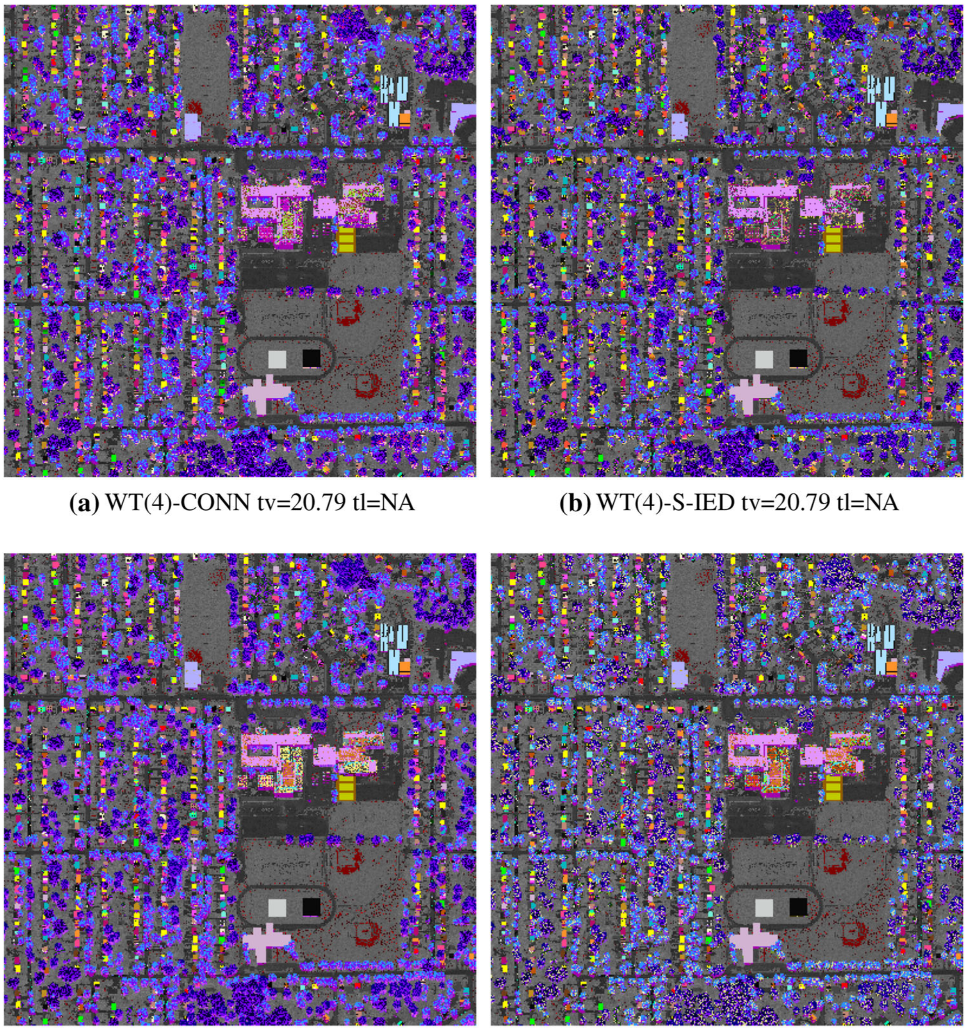
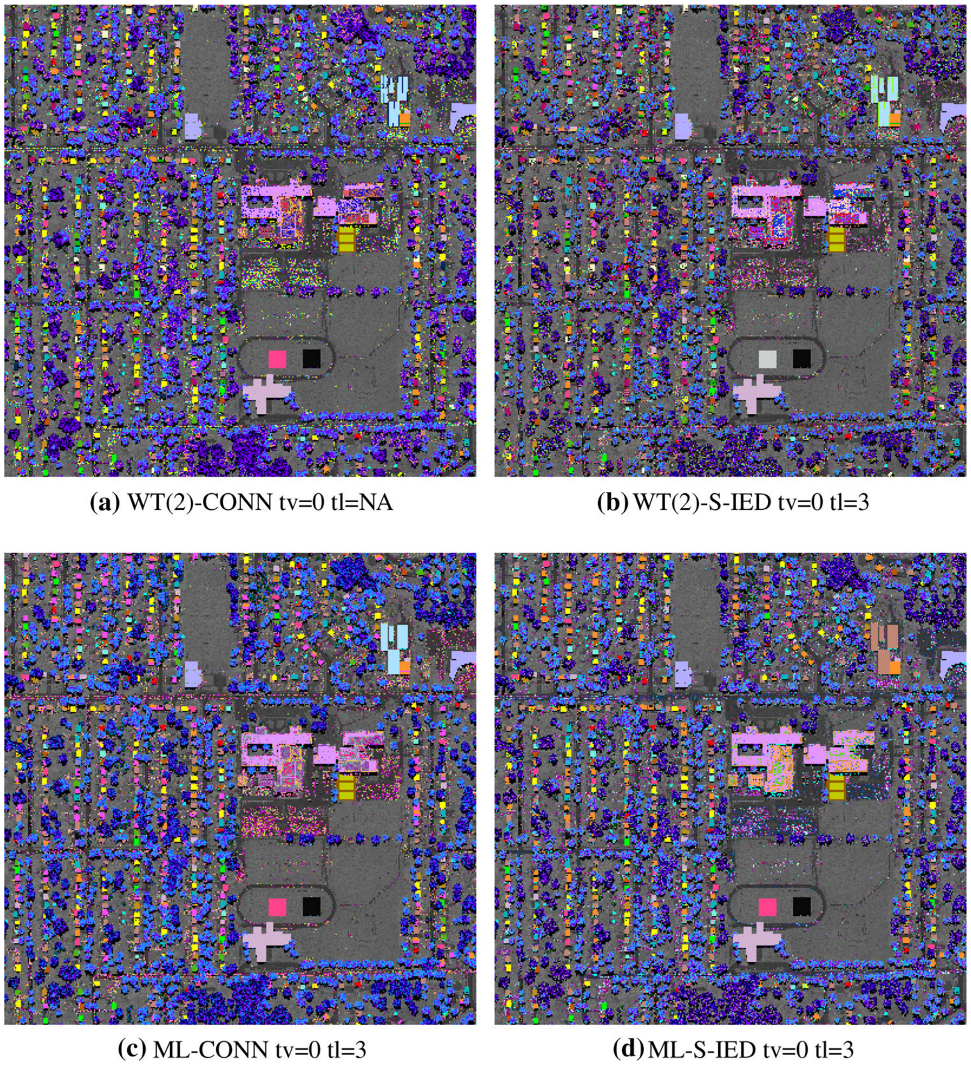

(a) WT(4)-CONN tv=20.79 tl=NA
(b) WT(4)-S-IED tv=20.79 tl=NA
(c) ML-CONN tv=20.79 tl=NA
(d) ML-S-IED tv=20.79 tl=NA

Fig. 9 Representative sample of the top automated clusterings, produced by segmenting the SOM and inheriting the prototype labels from the interactive segmentation as explained in Sect. 2.3. This compares the automated segmentations to the I-C (excluding the large classes K, T, V). Matching cluster labels are overlain on a gray scale version of the spatial image. The results were judged by visual inspection, which coincides with the ARI ranking with minor differences. The left and right columns show, respectively, results when the algorithms use the CONN and S-IED similarity for edge weighting. In all cases, CONN value (connectivity strength) thresholding is done with tv = 20.79 (mean of fourth-ranking connections), and no CONN length thresholding is applied. Top row: The best-performing case, WT(4) with CONN similarity (left), and its S-IED counterpart (right). For WT(4) CONN length thresholding with tl = 4 or tl = 3 is equally good in this case. Bottom row: The third-best clustering after the WT(x) family, by ML with CONN similarity (left) and S-IED counterpart (right). ML represents FG, LE, ML, which perform equally

yields markedly different behavior; IM's tendency to return many subclusters keeps its ARI agreement below 0.25 in all CONN thresholding schemes, while the other methods show different capacities to harness the effects of thresholding. For example, value thresholding tv = 20.79 dramatically improves WT(2) ARI performance (from 0.59 to 0.92). Similarly, length thresholding alone (tv = 0, tl = 3) increases the ARI score markedly. Applying both value and length thresholding causes the ARI decrease back to 0.58. All Walktrap parameterizations shown here behave similarly. Interestingly, the S-IED Walktrap variants perform best in the Vegetation Clusters with no thresholding. This can be explained by the fact that S-IED only inherits CONN sparsity but uses inverse Euclidean distance. Thresholding removes edges from the graph for which the CONN value is low. These tend to be edges between dissimilar nodes, whose elimination further diminishes the already low discrimination (as seen in Fig. 6) by Euclidean distance. IM with S-IED also performs significantly better with no CONN thresholding, not only by the ARI score but also by markedly less fragmentation (17 vs 32 clusters with tv = 0 vs other thresholding). Interestingly, ML behaves



(a) WT(2)-CONN tv=0 tl=NA
(b) WT(2)-S-IED tv=0 tl=3
(c) ML-CONN tv=0 tl=3
(d) ML-S-IED tv=0 tl=3

Fig. 10 A sample of the top automated clusterings produced by segmenting the SOM and inheriting the truth labels assigned to the prototypes as explained in Sect. 2.3. This compares the automated segmentations to the truth map (excluding the large classes K, T, V). Matching cluster labels are overlain on a gray scale version of the spatial image. The methods were judged by visual inspection, which coincides with the ARI ranking with minor differences. The left and right columns show, respectively, results when the algorithms use the CONN and S-IED similarity measure for edge weighting. In all cases, CONN value thresholding is done with tv = 20.79 (mean of fourth-ranking connections), and CONN length thresholding tl = 3 except for **a**. Top row: the best-performing case, WT(2) with CONN similarity (left). In this case, we show WT(2) with S-IED (right) for a different CONN length thresholding because the S-IED counterpart with matching thresholding is significantly worse. Bottom row: the third-best clustering after the WT(x) family, by ML with CONN similarity (left) and the S-IED counterpart (right)

the opposite way: It does better with no thresholding when using CONN; and better without when using S-IED. These observations lead us to believe CONN's values elucidate the vegetation more cleanly, whereas only CONN sparsity is needed to identify the small detailed clusters. This may be due to the high degree of CONN separation of the Small Clusters. Figure 11 provides a sense of the relative differences indicated by the ARI scores, in close-ups of representative details. The best-performing WT(4) clustering (second column) compares very well with both the truth image and the I-C in terms of localizing roof materials, trees, and the grass at the baseball diamond (cluster Y, rust color, matching the same cluster in the I-C). The third column portrays the ML clustering with S-IED (the measure most advantageous for ML) but with the same CONN thresholdings as for WT(4), which is not ideal for ML according to the ARI chart in Figs. 7 and 8. Compared to performance with best thresholding for ML (from Figs. 9 and 10), these are noticeably poorer matches.

Some clusters, commonly returned by all methods, exist in the I-C (and spectrally justified) but not in the truth map for reasons we noted above. In addition to the previously discussed baseball diamond (cluster Y), another readily identifiable example is the large light blue building toward

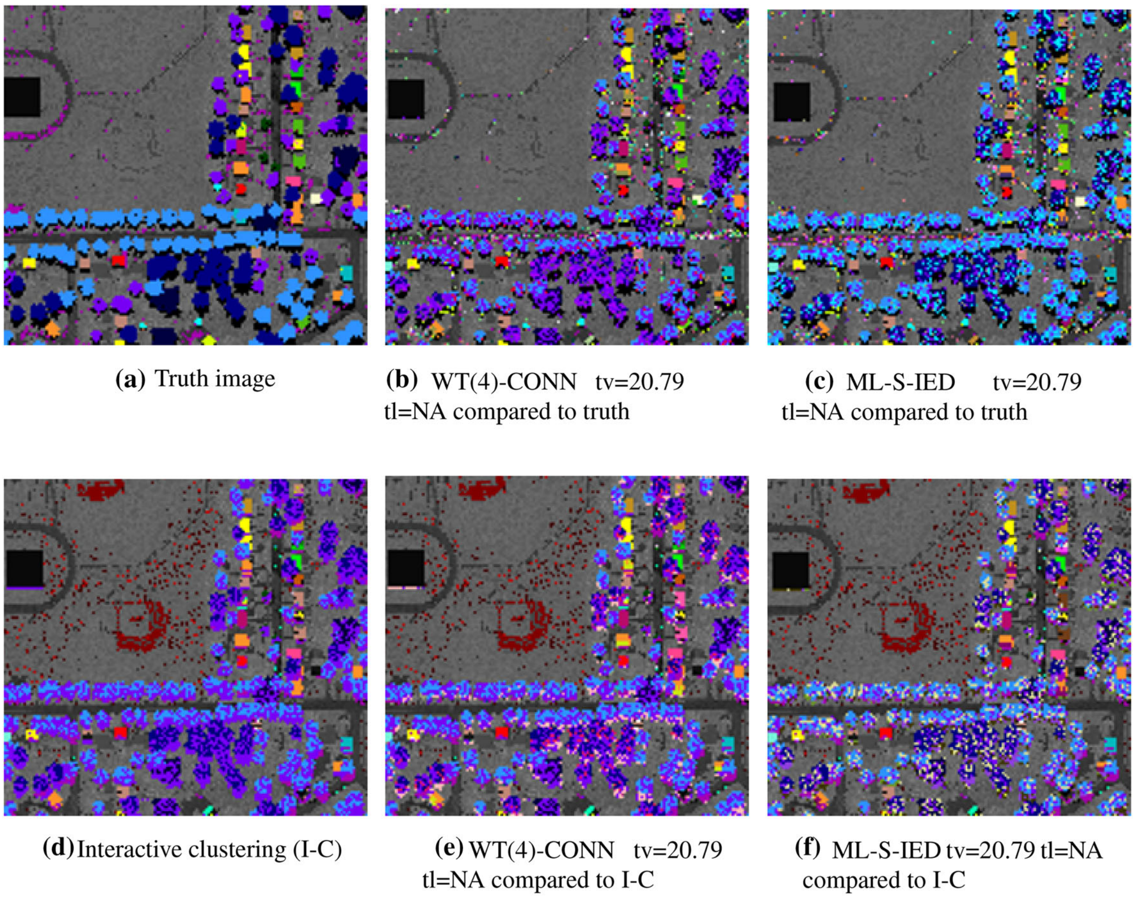

(a) Truth image
(b) WT(4)-CONN tv=20.79 tl=NA compared to truth
(c) ML-S-IED tv=20.79 tl=NA compared to truth
(d) Interactive clustering (I-C)
(e) WT(4)-CONN tv=20.79 tl=NA compared to I-C
(f) ML-S-IED tv=20.79 tl=NA compared to I-C

Fig. 11 Details from the lower right corner of the image, showing representative performance differences between the top-performing WT(4) with CONN similarity, and ML with S-IED, thresholdings as indicated. **a** The truth map; **b** WT(4) with CONN similarity and its best thresholding, as compared (reconciled) with the truth map; **c** ML with S-IED and the same thresholdings as for WT(4), as compared to the truth map. **d** I-C; **e** the same WT(4) clustering as in (**b**) but compared to the I-C; **f** the same ML clustering as in (**c**) but compared to the I-C

the upper right of the cluster map, a pool with glass cover labeled "glass". However, there must be other material (maybe metal) in the roof structure, which is marked by all clusterings as different and therefore mismatch the truth map. The `igraph` methods also discover common small classes in addition to those present in the I-C. All these lower ARI scores when, in fact, they indicate more precise clustering.

Despite their differences, all methods that use CONN or S-IED similarity localize the major structures (fields, trees, large buildings) and most smaller objects like various roofs, glass, very well. The differences in cluster delineation come down to the ability to discriminate more finely between spectrally similar materials. The small scale on which different materials alternate in this image generates a large number of mixed pixels causing unavoidable confusion at cluster boundaries. The main confusion caused by the labeling of g type pixels (whose spectra are inconsistent with vegetation) as vegetation unduly decreases the ARI score for the Vegetation Clusters. Given this and the noise, the relatively low ARI scores versus truth labels (around 50% for Small Clusters and 25% for the Vegetation Clusters) can be considered very good for this image. Visual impression confirms this. Further, all methods identify "extra" clusters" (consistent with discovery of the same by the I-C) that are spectrally justified but not distinguished in the truth map. While this again lowers the ARI scores, it attests to their discovery capabilities, and ARI values still provide relative merits.

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

3 Discussion, conclusion and outlook

Common to all 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 underperform those that use the CONN edge weighting. Considering that (under mild conditions, and assuming correct SOM learning) the CONN graph is the weighted masked Delaunay graph of the n-D data cloud [16, 31] where the weighting senses the "weak seams" in the manifold in both global and local relations, this is perhaps not surprising.

We conclude that the usual pairwise Euclidean distances of the 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 matrix produces results that approach the quality of interactive SOM clustering. Additionally, automatically determined, data-dependent thresholdings of the CONN edge weightings reveal further benefits that help increase modularity or the probability of visiting relevant subcommunities (by Walktrap) and thereby increase performance. The automated approaches can utilize more nuanced information from the CONN matrix than the human operator, and do not get tired, which can result in more complete labeling of the SOM as is the case in the Megascene example. With the automated methods the SOM segmentation takes negligible time, $\ll 1$ s, eliminating the bottleneck that currently prohibits high-throughput analyses.

In this paper, we cluster somewhat complex 6-D synthetic data, and realistically complex and noisy, simulated hyperspectral image data whose cluster structure can be verified against known templates of truth labels, and against independently produced interactive clustering. These data sets facilitate experimentation with a wider set of graph segmentation algorithms for systematic and more thorough isolation of the relative advantages. Walktrap with 4 or 2 steps works best for the large hyperspectral data in this study. However, 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, parametrization, similarity measure (CONN or S-IED), and CONN thresholding may be best. This work is the first step in charting the behaviors in this combined parameter space. Follow-up work will extend this to other types of large, complex data. Finally, we note that SOM learning—the key to all of this—is a lengthy iterative process. To accelerate it, parallel hardware implementations can be used (see, e.g., [14]) to bring the overall processing time of cluster discovery to a level that scales with Big Data.

Acknowledgements We thank Dr. John Kerekes, Rochester Institute of Technology, for the RIT synthetic hyperspectral image cube and accompanying truth labels. This project was partially supported by a North American ALMA Development Cycle 5 Study Program, administered by the National Radio Astronomy Observatory, with the consent of the US National Science Foundation.

References

1. Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 10:P10008
2. Brugger D, Bogdan M, Rosenstiel W (2008) Automatic cluster detection in Kohonen's SOM. IEEE Trans Neural Netw 19(3):442–459
3. Clauset A, Newman MEJ, Moore C (2004) Finding community structure in very large networks. Phys Rev E 70:066111. https://doi.org/10.1103/PhysRevE.70.066111
4. Cottrell M, de Bodt E (1996) A Kohonen map representation to avoid misleading interpretations. In: Proceedings of the 4th European symposium on artificial neural networks (ESANN'96), D-Facto, Bruges, pp 103–110
5. Cottrell M, Rousset P (1997) The Kohonen algorithm: a powerful tool for analyzing and representing multidimensional quantitative and qualitative data. In: Proceedings of the international work-conference on artificial neural networks, pp 861–871
6. Csardi G, Nepusz T (2006) The igraph software package for complex network research. InterJournal Complex Syst 1695:1–9
7. Danon L, Daz-Guilera A, Duch J, Arenas A (2005) Comparing community structure identification. J Stat Mech Theory Exp 09:P09008
8. Fortunato S (2010) Community detection in graphs. Phys Rep 486(3–5):75–174. https://doi.org/10.1016/j.physrep.2009.11.002
9. Goncalves M, Netto M, Costa JAF (2008) A new method for unsupervised classification of remotely sensed images using Kohonen Self-Organizing Maps and agglomeration hierarchical clustering methods. Int J Remote Sens 11(29):3171–3207
10. Hamel L, Brown C (2011) Improved interpretability of the unified distance matrix with connected components. In: Proceedings of the 7th international conference on data mining (DMIN'11). CSREA Press, Las Vegas, pp 338–343
11. Hubert L, Arabie P (1985) Comparing partitions. J Classif 2(1):193–218. https://doi.org/10.1007/BF01908075
12. Ientilucci E, Brown S (2003) Advances in wide-area hyperspectral image simulation. Proc SPIE 5075:110–121
13. Kohonen T (1988) Self-organization and associative memory. Springer, New York
14. Lachmair J, Merényi E, Porrmann M, Rückert U (2013) A reconfigurable neuroprocessor for self-organizing feature maps. Neurocomputing 112:189–199
15. Liao W, Chen H, Yang Q, Lei X (2008) Analysis of fMRI data using improved self-organizing mapping and spatio-temporal metric hierarchical clustering. IEEE Trans Med Imaging 27(10):1472–1482
16. Martinetz T, Schulten K (1994) Topology representing networks. Neural Netw 7(3):507–522

17. Mendenhall M, Merényi E (2009) On the evaluation of synthetic hyperspectral imagery. In: Proceedings of the first workshop on hyperspectral image and signal processing: evolution in remote sensing (WHISPERS 2009), Grenoble. http://www.ece.rice.edu/~erzsebet/papers/. ISBN 978-1-4244-4687-2
18. Merényi E, Jain A, Villmann T (2007) Explicit magnification control of self-organizing maps for "forbidden" data. IEEE Trans Neural Netw 18(3):786–797
19. Merényi E, Taşdemir K, Farrand W (2008) Intelligent information extraction to aid science decision making in autonomous space exploration. In: Fink W (ed) Proceedings of the DSS08 SPIE defense and security symposium, space exploration technologies, vol 6960. SPIE, Orlando, p 69600M. http://scitation.aip.org/dbt/dbt.jsp?KEY=PSISDG&Volume=6960&Issue=1.
20. Merényi E, Taşdemir K, Zhang L (2009) Learning highly structured manifolds: harnessing the power of SOMs. In: Biehl M, Hammer B, Verleysen M, Villmann T (eds) Similarity-based clustering, vol 5400. Lecture notes in computer science. Springer, Berlin, Heidelberg, pp 138–168
21. Merényi E, Taylor J (2017) SOM-empowered graph segmentation for fast automatic clustering of large and complex data. In: 12th International workshop on self-organizing maps and learning vector quantization, clustering and data visualization (WSOM+ 2017), pp 1–9
22. Merényi E, Taylor J, Isella A (2016) Deep data: discovery and visualization. Application to hyperspectral ALMA imagery. Proc Int Astron Union 12(S325):281–290. https://doi.org/10.1017/S1743921317000175
23. Merényi E, Taylor J, Isella A (2016) Mining complex hyperspectral ALMA cubes for structure with neural machine learning. In: 2016 IEEE symposium series on computational intelligence (SSCI), pp 1–9. https://doi.org/10.1109/SSCI.2016.7849952
24. Murtagh F (1995) Interpreting the Kohonen self-organizing feature map using contiguity-constrained clustering. Pattern Recognit Lett 16(4):399–408
25. Newman MEJ (2004) Fast algorithm for detecting community structure in networks. Phys Rev E 69:066133. https://doi.org/10.1103/PhysRevE.69.066133
26. Newman MEJ (2006) Finding community structure in networks using the eigenvectors of matrices. Phys Rev E 74:036104. https://doi.org/10.1103/PhysRevE.74.036104
27. Pons P, Latapy M (2005) Computing communities in large networks using random walks. In: Proceedings of the 20th international conference on computer and information sciences, ISCIS'05. Springer, Berlin, Heidelberg, pp 284–293. https://doi.org/10.1007/11569596_31
28. Rosvall M, Bergstrom CT (2008) Maps of random walks on complex networks reveal community structure. Proc Natl Acad Sci 105(4):1118–1123. https://doi.org/10.1073/pnas.0706851105
29. Schott J, Brown S, Raqueo R, Gross H, Robinson G (1999) An advanced synthetic image generation model and its application to multi/hyperspectral algorithm development. Can J Remote Sens 25(2):99–111
30. Taşdemir K (2011) Spectral clustering as an automated SOM segmentation Tool. In: Proceedings of the workshop on self-organizing maps (WSOM 2011), pp 71–78
31. Taşdemir K, Merényi E (2009) Exploiting data topology in visualization and clustering of Self-Organizing Maps. IEEE Trans Neural Netw 20(4):549–562
32. Taşdemir K, Merényi E (2011) A validity index for prototype based clustering of data sets with complex structures. IEEE Trans Syst Man Cybern Part B 41(4):1039–1053. https://doi.org/10.1109/TSMCB.2010.2104319
33. Taşdemir K, Milenov P, Tapsall B (2011) Topology-based hierarchical clustering of self-organizing maps. IEEE Trans Neural Netw 22(3):474–485
34. Ultsch A (2005) Clustering with SOM: U*c. In: Proceedings of the 5th workshop on self-organizing maps (WSOM 2005), Paris, pp 75–82
35. Vesanto J, Alhoniemi E (2000) Clustering of the self-organizing map. IEEE Trans Neural Netw 11(3):586–600
36. Ward JH (1963) Hierarchical grouping to optimize an objective function. J Am Stat Assoc 58(301):236–244. https://doi.org/10.1080/01621459.1963.10500845