An Efficient Formulation of the Improved Visual Assessment of Cluster Tendency (iVAT) Algorithm

Timothy C. Havens, *Senior Member, IEEE*, and James C. Bezdek, *Fellow, IEEE*

***Abstract*—The VAT algorithm is a visual method for determining the possible number of clusters in, or the cluster tendency of, a set of objects. The *improved* VAT (iVAT) algorithm uses a graph-theoretic distance transform to improve the effectiveness of the VAT algorithm for "tough" cases where VAT fails to accurately show the cluster tendency. In this paper we present an efficient formulation of the iVAT algorithm which reduces the computational complexity of the iVAT algorithm from $O(N^3)$ to $O(N^2)$. We also prove a direct relationship between the VAT image and the iVAT image produced by our efficient formulation. We conclude with three examples displaying clustering tendencies in three of the Karypis data sets that illustrate the improvement offered by the iVAT transformation. We also provide a comparison of iVAT images to those produced by the *Reverse Cuthill-Mckee* (RCM) algorithm; our examples suggest that iVAT is superior to the RCM method of display.**

***Index Terms*—clustering, cluster tendency, visualization, VAT**

I. Introduction

Consider a set of objects $\mathbf{O} = \{o_1, \ldots, o_N\}$, where the objects could be the set of all bass guitars manufactured before 1970, celebrity cats, genes in a microarray experiment, the radar-signatures of explosive hazards encountered in Afghanistan, pelagic fish in the Northern hemisphere, etc. Clustering is the process by which the natural groupings in $\mathbf{O}$ are determined, such that the objects in each group exhibit more similarity to one another than to objects in other groups. Clustering has also been called exploratory data analysis, unsupervised learning, numerical taxonomy, typology, and partitioning [1]. Some good general references on clustering include [1–8].

In conventional (object-data) cluster analysis, the objects are separated into groups according to their features $\vec{x}_i \in \mathbb{R}^p$, where $\vec{x}_i$ is the p-dimensional feature vector of the ith object and each element of $\vec{x}_i$ is a numerical feature such as weight, height, gene-expression, or voltage. An alternative form of data is *relational* data, where only the relationship between pairs of objects is known. This type of data is especially prevalent in document-analysis and bioinformatics. Relational data typically consists of the N^2 values of a square dissimilarity matrix $\mathbf{D}$, where $D_{ij} = d(o_i, o_j)$ is the pairwise dissimilarity (or distance) between objects o_i and o_j. For instance, numerical data X can always be converted to $\mathbf{D}$ by $D_{ij} = \|\vec{x}_i - \vec{x}_j\|$ (any vector norm on $\mathbb{R}^p$). There are, however, similarity and dissimilarity relational data sets that do not begin as numerical object data; for these, there is no choice but to use a relational algorithm. Hence, relational data represent the "most general" form of input data.

Although clustering is typically thought of as only the act of separating objects into the proper groups, cluster analysis actually consists of three concise questions: i) *cluster tendency*—how many clusters are there? ii) *partitioning*—which objects belong to which cluster and to what degree? iii) *cluster validity*—are the partitions "good"? Because most clustering algorithms require the number of clusters as an input, cluster tendency is an important problem in cluster analysis. The *Visual Assessment of cluster Tendency* (VAT) algorithm [9] addresses this question by reordering the dissimilarity matrix $\mathbf{D}$ so that, ideally, the number of clusters is displayed as the number of "dark blocks" along the diagonal. Recently, an *improved* VAT (iVAT) algorithm was proposed [10] which first transforms $\mathbf{D}$ using a graph-theoretic distance transform. Then VAT is used on the transformed dissimilarity matrix. Thus, the iVAT algorithm can be interpreted as a feature extraction technique. The examples shown in [10] suggest that iVAT significantly improves the contrast of the "dark blocks" in most VAT images, resulting in an easier determination of the cluster tendency.

In this paper, we present a efficient formulation of the iVAT algorithm which significantly reduces its computational complexity. Our iVAT implementation begins by finding the VAT reordered dissimilarity matrix and then performs a distance transform on this matrix. We show that the resulting matrix is an iVAT image. Section II describes the efficient iVAT algorithm, some examples are presented and comparisons are made in Section III, and Section IV presents a short discussion and some ideas for future research.

A. Matrix Reordering Methods

It is considered that Petrie, in 1899, was the first to use matrix permutation for discovering trends in measurement data [11]. However, in 1909, Czekanowski apparently presented the first method for clustering in dissimilarity data using a visual approach [12]. Czekanowski reordered, by hand, a 13×13 dissimilarity matrix that represented the average difference in various skulls, showing that the skulls roughly clustered into two distinct groups. These pieces of

This material is based upon work supported by the National Science Foundation under Grant #1019343 to the Computing Research Association for the CI Fellows Project.

T.C. Havens is with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824, USA (corresponding author: phone: 231-360-8444 email: havenst@gmail.com).

J.C. Bezdek is with the Department of Electrical and Computer Engineering, University of Missouri, Columbia, MO 65211, USA (email: jcbezdek@gmail.com).

history and other important milestones on the use of heat maps to visualize clusters are described in detail in [13].

Tryon [14] later presented another method for visual clustering in dissimilarity data. Here is a rough description of his method; (i) plot a graph of each row in the data—a matrix of pair-wise correlation coefficients, (ii) visually aggregate subsets of the graphs into clusters, (iii) find the mean profile (a prototype graph representing the elements of a group) for each cluster of correlation profiles, and (iv) present the final results as a set of clustered profile graphs with their prototypes. This procedure—almost 70 years old—contains most of the elements of the current work on visual clustering: create a visual representation of $\mathbf{D}$, reorder it to $\mathbf{D}^*$ (Tryon did this implicitly, he did not construct an explicit reordering of $\mathbf{D}$), create a visual representation $\mathbf{D}^*$, and finally, extract clusters from $\mathbf{D}^*$ using the visual evidence. Tryon did this by hand in 1939 for a 20×20 data set collected at the University of California, Berkeley. For tiny data sets, methods such as these are useful. But for the data sets typically encountered today, automation is essential.

Cattell [15] introduced the idea of visual representation of $\mathbf{D}^*$ by an image. He presented an image $I(\mathbf{D}^*)$ of $\mathbf{D}^*$ created by hand-shading the pixels of a matrix $\mathbf{D}$ with one of three "intensities". Reordering was done by hand. In 1957, Sneath [16] was the first to introduce an element of modern computing into this procedure; he constructed the matrix $\mathbf{D}$ with a computer, but the matrix $\mathbf{D}^*$ and subsequent image of it $I(\mathbf{D}^*)$ were still built by hand.

Subsequent refinements of Cattell's and Sneath's ideas followed the general evolution of computers themselves. Floodgate and Hayes [17] presented a hand rendered image similar to Sneath's, but reordering of $\mathbf{D}$ was done computationally using single-linkage clustering. Apparently, Ling [18] was the first to automate the creation of the image $I(\mathbf{D}^*)$ of $\mathbf{D}^*$ with an algorithm called SHADE, which was used after application of the complete linkage hierarchical clustering scheme and served as an alternative to visual displays of hierarchically nested clusters via the standard dendrogram. SHADE used 15 level halftone intensities (created by overstriking standard printed characters) to approximate a digital representation of the lower triangular part of the reordered dissimilarity matrix. SHADE apparently represents the first completely automated approach to finding $\mathbf{D}^*$ and viewing $I(\mathbf{D}^*)$. However, SHADE is computationally expensive because it is tied to a specific clustering method, complete linkage, and, of course, the creation of images by computer is far more advanced than the method developed by Ling.

Closely related to SHADE, but presented more in the spirit of finding rather than displaying clusters found with a relational clustering algorithm, is the "graphical method of shading" described by Johnson and Wichern in [19]. They provide this informal description: (i) arrange the pair-wise distances between points in the data into several classes of 15 or fewer, based on their magnitudes, (ii) replace all distances in each class by a common symbol with a certain shade of gray, (iii) reorganize the distance matrix so that items with common symbols appear in contiguous locations along the main diagonal (darker symbols correspond to smaller distances), and (iv) identify groups of similar items by the corresponding patches of dark shadings. A more formal approach to this problem is the work of Tran-Luu [20], who proposed reordering the data into an "acceptable" block form based on optimizing several mathematical criteria of image "blockiness". The reordered matrix is then imaged and the number of clusters is deduced visually by a human observer. However, these methods require optimizing an objective function; thus, success often comes at a high computational cost.

Similarity-based intensity images, formed using kernel functions, have been used in [21] and [22] to provide guidance in determining the number of clusters (tendency assessment, in spirit of the VAT and iVAT algorithms), but no useful ordering scheme is offered there to facilitate the approach.

Graph-based methods for reordering matrices are numerous. Perhaps the most simple are breadth-first search and depth-first search [23]. These search algorithms, which also work with weighted graphs, add vertexes (objects) to the matrix image in the order in which vertices are searched. Although these methods share the low time complexity of VAT, $O(N^2)$, the results are known to be very susceptible to initialization.

Other methods only work with un-weighted connected graphs. A good general reference on many of these methods is [24], including descriptions and comparisons. These methods include the degree ordering, *Reverse Cuthill-Mckee* (RCM) [25], and King's [26] algorithms. The Sloan algorithm [27] is a recent advance that seems better than these algorithms because of its lower time complexity. All these algorithms attempt to move non-zero elements closer to the diagonal. This is essentially the goal of cluster tendency visualization. However, they only work with binary connection matrices. Dissimilarity matrices can easily be converted to binary matrices by thresholding. But the results are very sensitive to this threshold value and, for this reason, these algorithms are not good methods for determining cluster tendency in weighted graphs or proximity (dissimilarity) data.

A recent innovation is spectral ordering, in which the sum of all edge lengths used is minimized [23]. Spectral ordering sorts the matrix entries according to an Eigen-based decomposition of the graph's Laplacian matrix. This method is shown to be effective and stable, but suffers from very high computational complexity, $O(N^6)$ for the data we examine in this paper.

B. VAT and iVAT

The VAT algorithm displays an image of reordered and scaled dissimilarity data [9]. Each pixel of the grayscale VAT image $I(\mathbf{D}^*)$ displays the scaled dissimilarity value between two objects. White pixels represent high dissimilarity, while black represents low dissimilarity. Each object is exactly

similar with itself, which results in zero-valued (black) diagonal elements in $I(\mathbf{D}^*)$. The off-diagonal elements of $I(\mathbf{D}^*)$ are scaled to the range $[0, 1]$. A dark block along the diagonal of $I(\mathbf{D}^*)$ is a sub-matrix of "similarly small" dissimilarity values; hence, the dark block represents a cluster of objects that are relatively similar to each other. Thus, cluster tendency is shown by the number of dark blocks along the diagonal of the VAT image.

The VAT algorithm is based on (but not identical to) Prim's algorithm [28] for finding the *minimum spanning tree* (MST) of a weighted connected graph [9]. Algorithm 1 lists the steps of the VAT algorithm. Note that the objective of finding the MST in $\mathbf{D}$ is to obtain the sequence of indices in which edges are added to the tree. Subsequently, the indices are used to effect the reordering of $\mathbf{D}$; in particular, the MST is not cut, as in single linkage clustering, to find partitions of the data. The resulting VAT-reordered dissimilarity matrix $\mathbf{D}^*$ can be normalized and mapped to a gray-scale image with black representing the minimum dissimilarity and white the maximum.

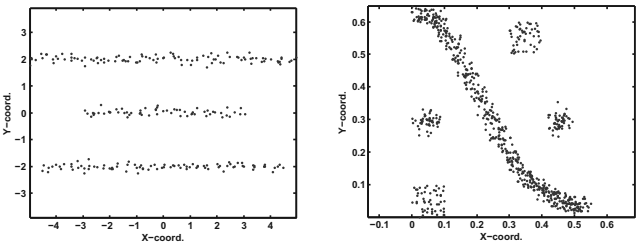
Algorithm 1: VAT Reordering Algorithm [9]

Input: $\mathbf{D}$ — $N \times N$ dissimilarity matrix
Data: $\mathbf{K} = \{1, 2, \ldots, N\}$; $\mathbf{I} = \mathbf{J} = \emptyset$;
$P = (0, 0, \ldots, 0)$.
Select $(i, j) \in \arg\max_{p \in \mathbf{K}, q \in \mathbf{K}} D_{pq}$.
Set $P(1) = i$; $\mathbf{I} = \{i\}$; and $\mathbf{J} = \mathbf{K} - \{i\}$.
for $r = 2, \ldots, N$ **do**
  Select $(i, j) \in \arg\min_{p \in \mathbf{I}, q \in \mathbf{J}} D_{pq}$.
  Set $P(r) = j$; Replace $\mathbf{I} \leftarrow \mathbf{I} \cup \{j\}$ and $\mathbf{J} \leftarrow \mathbf{J} - \{j\}$.
Obtain the ordered dissimilarity matrix $\mathbf{D}^*$ using the ordering array P as: $D^*_{pq} = D_{P(p),P(q)}$, for $1 \leq p, q \leq N$.
Output: Reordered dissimilarity $\mathbf{D}^*$

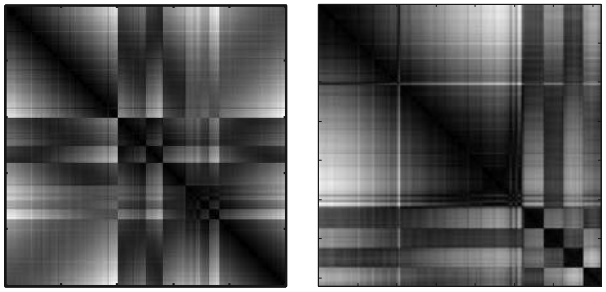
Reference [10] proposed an *improved* VAT (iVAT) algorithm that uses a path-based distance measure from [29]. Consider $\mathbf{D}$ to represent the weights of the edges of a fully-connected graph. The path-based distance is defined as

$$D'_{ij} = \min_{p \in P_{ij}} \max_{1 \leq h < |p|} D_{p[h]p[h+1]}, \tag{1}$$

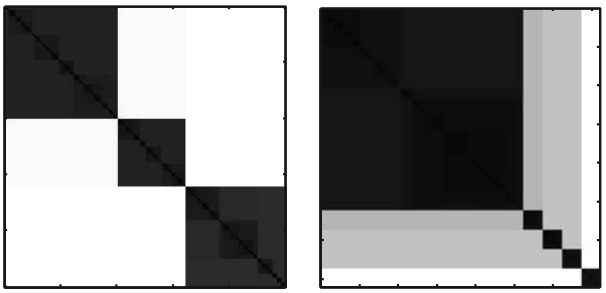
where $p \in P_{ij}$ is an acyclic path in the set of all acyclic paths between vertex i (o_i) and vertex j (o_j), $p[h]$ is the index of the hth vertex along path p, and $|p|$ is the number of vertices along the path. Hence, $D_{p[h]p[h+1]}$ is the weight of the hth edge along path p. Essentially the cost of each path p, for the distance in Eq.(1), is the maximum weight of its $|p|$ edges. The distance between i and j is the minimum-cost path in P_{ij}. In the spirit of other algorithms, such as the dimensionality reduction method in [30], that perform a distance transform as a preprocessing step, the authors of the original iVAT paper [10] first transform $\mathbf{D}$ into $\mathbf{D}'$ using a shortest-path algorithm, where the cost is computed by (1); then they use VAT on the transformed dissimilarity matrix.



(a) 3 Lines object data (b) Boxes and Stripe object data



(c) 3 Lines VAT image - $I(\mathbf{D}^*)$ (d) Boxes and Stripe VAT image - $I(\mathbf{D}^*)$



(e) 3 Lines iVAT image - $I(\mathbf{D}'^*)$ (f) Boxes and Stripe iVAT image - $I(\mathbf{D}'^*)$

Fig. 1: VAT and iVAT images of dissimilarity data where VAT "fails"

The iVAT images show considerable improvement over VAT images in showing the cluster tendency for "tough" cases.

Figure 1 illustrates the ability of iVAT to show correct cluster tendency for two cases where VAT "fails". Views (a,b) show object data upon which the Euclidean norm was used to compute dissimilarity data $\mathbf{D}$. Views (c,d) show the respective VAT images of the *3 Lines* and *Boxes and Stripe* data. The VAT image of the *3 Lines* data clearly does not show the preferable tendency of 3 clusters. Although the VAT image of the *Boxes and Stripe* data shows the four tightly grouped clusters as 4 distinct dark blocks in the lower-right of the image, the large wavy cluster (the stripe) is not distinctly shown. In both cases, the iVAT images, shown in views (e,f), clearly indicate the preferred cluster tendency—3 clusters in the *3 Lines* data and 5 clusters in the *Boxes and Stripe* data.

Computing $\mathbf{D}'$ directly from (1) can be thought of as a shortest path problem. The Floyd-Warshall algorithm [31] is an algorithm that solves this problem for all N^2 pairs of lowest-cost paths in a connected graph with N nodes. The Floyd-Warshall algorithm has a complexity of $O(N^3)$. The

complexity of the VAT algorithm is $O(N^2)$; thus, the total complexity of iVAT as proposed in [10] is $O(N^3 + N^2) = O(N^3)$.

In this paper we propose an efficient formulation which reduces the computational complexity of the iVAT distance transform in Eq.(1) to $O(N^2)$. Thus the total complexity of the iVAT formulation presented here is $O(N^2 + N^2) = O(N^2)$. In Section II we show that i) the iVAT dissimilarity matrix $\mathbf{D}'$ can be computed directly from the VAT-reordered data $\mathbf{D}^*$ (see Algorithm 2) and ii) the matrix $\mathbf{D}'$ computed by our algorithm is already in VAT-order. We denote the iVAT transformed dissimilarity matrix $\mathbf{D}'$ as $\mathbf{D}'^*$ to indicate that it is a VAT-ordered matrix.

II. Efficient iVAT

Algorithm 2 outlines the efficient formulation of iVAT. In contrast to the iVAT formulation in [10], we start by applying VAT to the input dissimilarity matrix, and then transform the VAT-reordered dissimilarity data into the iVAT image with our recursive algorithm, which we will call efiVAT. The transformation sequence for our efficient iVAT is thus VAT($\mathbf{D}$) = $\mathbf{D}^*$ $\rightarrow$ efiVAT($\mathbf{D}^*$) = $\mathbf{D}'^*$, whereas the computational sequence for the original (non-recursive) version of iVAT is $\mathbf{D} \rightarrow \mathbf{D}' \rightarrow$ VAT($\mathbf{D}'$) = D'^*.

Line 1 of Algorithm 2 requires $(r-1)$ comparisons and Line 3 requires $(r-2)$ comparisons, for a total of $(2r-3)$ operations.[1] The total number of operations in Algorithm 2 is thus $(2N^2 - 3N)$, which is $O(N^2)$ complexity. We were able to reduce the complexity to $O(N^2)$ by using VAT itself as a preprocessing step. In contrast to the iVAT formulation in [10], we start with VAT and then transform the VAT-reordered dissimilarity data into the iVAT image with our algorithm.

Algorithm 2: Efficient calculation of iVAT image

Input: $\mathbf{D}^*$ - VAT-reordered dissimilarity matrix
Data: $\mathbf{D}'^* = [0]^{N \times N}$
for $r = 2, \ldots, N$ **do**
1 $\quad j = \arg\min_{k=1,\ldots,r-1} D^*_{rk}$
2 $\quad D'^*_{rc} = D^*_{rc}, c = j$
3 $\quad D'^*_{rc} = \max\left\{D^*_{rj}, D'^*_{jc}\right\}, c = 1, \ldots, r-1, c \neq j$
$\mathbf{D}'^*$ is symmetric, thus $D'^*_{rc} = D'^*_{cr}$.

The lemmas presented here show that the iVAT image can be recursively computed from the VAT image using Algorithm 2 and, also, suggest that our algorithm is an example of dynamic programming.

[1]We would like to note that the number of operations in the VAT$\rightarrow$iVAT pair can be further reduced by noticing that there is a coupling between our VAT and iVAT algorithms, as shown in Algorithms 1 and 2. The object indexed j chosen in line 1 of our efficient iVAT is exactly the same object indexed i in line 3 of VAT. Thus, one can store these indices from VAT and reuse them in iVAT, essentially eliminating the argmin operation in line 1 of iVAT. This does not change the overall complexity of our efficient iVAT implementation; it is still quadratic. However, our empirical observations on reusing this index show a reduced run-time by a factor of 2-3 for big dissimilarity matrices.

Without loss of generality, let the objects in any VAT-reordered object data $\mathbf{O}^*$ also represent the vertices of a fully connected graph, where the individual vertices are denoted as o^*_i, $i = 1, \ldots, N$. For ease of notation, I denote the vertices simply by the objects' indexes, $i = 1, \ldots, N$. The edge weights of this graph are either given, if the input data are not derived from object vectors, or computed by your chosen vector norm or dissimilarity function d,

$$\text{edge weight}_{ij} = d(i,j) = D^*_{ij}. \tag{2}$$

Then,

$$d_{min}(I,J) = \min_{\forall i \in I, \forall j \in J} D^*_{ij} \tag{3}$$

Lemma II.1. *Consider a vertex $k, 1 < k < N$, and the sets of vertices, $I = \{1, \ldots, k-1\}$ and $J = \{k+1, \ldots, N\}$. Then,*

$$d_{min}(I,k) \leq d_{min}(I,J). \tag{4}$$

Proof: Recall that VAT is a special case of Prim's algorithm and, thus, computes the *minimum-spanning-tree* (MST) of a set of vertices (objects) by adding the vertex which is closest to the already ordered vertices. By the definition of VAT, (4) is true. ■

Remark 1. In the case where each edge has a unique weight, then $d_{min}(I,k) < d_{min}(I,J)$ can be shown to be true [32].

The next lemma proves that line 2 in Algorithm 2 is valid. Note that $\mathbf{D}^*$ and $\mathbf{D}'$ are symmetric distance matrices.

Lemma II.2. *Consider the vertices $k, 1 < k \leq N$, and $l, 1 \leq l < k$, and the sets of vertices $I = \{1, \ldots, k-1\}$ and $J = \{k+1, \ldots, N\}$, where $J = \emptyset$ if $k = N$. If*

$$D^*_{kl} = d_{min}(I,k), \tag{5}$$

then the path-based distance (1) between k and l is

$$D'_{kl} = D^*_{kl}. \tag{6}$$

Proof: Lemma II.1 shows that $d_{min}(I,k) \leq d_{min}(I,J)$, which can be extended to $D^*_{kl} = d_{min}(I,k) \leq d_{min}(I,J)$. Thus,

$$D^*_{kl} \leq \min_p \max_{1 \leq h < |p|} D_{p[h]p[h+1]}, \tag{7}$$

for all paths $p \in P_{kl}$ that include a vertex in J. Equation (5) shows that

$$D^*_{kl} = \min_p \max_{1 \leq h < |p|} D_{p[h]p[h+1]}, \tag{8}$$

for all paths $p \in P_{kl}$ that include a vertex in I. Thus, $D'_{kl} = D^*_{kl}$.

Finally, if we consider (8) for the special case of $k = N$, it is easy to see that the lemma holds true. ■

Remark 2. Notice that D^*_{kl} in Lemma II.2 is the weight of the MST edge that connects vertex k to the sub-tree I.

The next lemma proves that line 3 of our algorithm is valid.

Lemma II.3. *Consider the vertices $k, 1 < k \leq N$, and $l, 1 \leq l < k$, and the sets of vertices $I = \{1, \ldots, k-1\}$ and $J = \{k+1, \ldots, N\}$, where $J = \emptyset$ if $k = N$. If*

$$D^*_{kl} > d_{min}(I, k) \tag{9}$$

and

$$s = \arg\min_{1 \leq t < k} D^*_{kt}, \tag{10}$$

then the path-based distance (1) between k and l,

$$D'_{kl} = \max\{D^*_{ks}, D'_{sl}\}. \tag{11}$$

Proof: Lemma II.3 is proved by showing that the path $p \in P_{kl}$ that produces the minimum $\max_{1 \leq h < |p|} D^*_{p[h]p[h+1]}$ is the path along the MST edges between vertices k and l. Lemma II.2 shows that $D^*_{ks} \leq d_{min}(I, J)$. Thus, all paths $p \in P_{kI}$ through the vertices J have an edge with a weight $\geq D^*_{ks}$. In other words, D^*_{ks} is the least costly path from vertex k to the sub-tree I, where $l \in I$. By definition, D'_{sl} is the value of the maximum edge weight along the least costly path from s to l, thus $D'_{kl} = \max\{D^*_{ks}, D'_{sl}\}$. ■

Remark 3. Notice that D^*_{ks} is the weight of the $(k-1)$th edge added the MST by Prim's algorithm. Additionally, it can be shown that D'_{sl} is the weight of the $(l-1)$th edge added to the MST. Thus, all the distance values in $\mathbf{D}'$ are the weights of the maximum MST edge that is traversed between vertices k and l. This logic can also be extended to show that the path p which produces each value of $\mathbf{D}'$ is the path along the MST.

Equations (8) and (11) are applied recursively, starting with $k = 2$, to calculate $\mathbf{D}'$ from $\mathbf{D}^*$. Now we show that the result of this recursive calculation is an iVAT image, $\mathbf{D}' = \mathbf{D}'^*$.

The properties of a VAT image are:

1) The first vertex in the ordering is one of the set of vertices that satisfy
$$k = \arg\max_{\forall i} \left\{ \max_{\forall j} D_{ij} \right\}. \tag{12}$$
2) The ordering of the vertices is that which could be computed using Prim's algorithm.

First, we show that $\mathbf{D}'$ satisfies the first VAT property by showing that the first row in $\mathbf{D}'$—the distances between vertex 1 and all other vertices—contains an element that is the maximum of $\mathbf{D}'$.

Lemma II.4. $\mathbf{D}'$ *satisfies*

$$\max_{\forall i} D'_{1i} = \max_{\forall i,j} D'_{ij}. \tag{13}$$

Proof: The path-based distance D'_{1i} is the value of the maximum weighted edge along the MST path between vertices 1 and i. Hence, $\max_{\forall i} D'_{1i}$ is the value of the maximum weighted MST edge because all MST edges are traversed on at least one path $p \in P^{MST}_{1,\forall j}$, where $P^{MST}_{1,\forall j}$ denotes all possible paths between vertex 1 and all other vertices that are on the MST. Additionally, all the elements of $\mathbf{D}'$ are MST edge weights. Thus, $\max_{\forall i} D'_{1i} = \max_{\forall i,j} D'_{ij}$. ■

Remark 4. An interesting consequence of Lemma II.4 is that it can be simply modified to show that *any* vertex can be chosen as the initial vertex in iVAT.

Next, we show that $\mathbf{D}'$ satisfies the second VAT property by first defining

$$d'_{min}(I, J) = \min_{\forall i \in I, \forall j \in J} D'_{ij}. \tag{14}$$

Lemma II.5. *Consider the vertex $1 < k < N$ and the sets of vertices, $I = \{1, \ldots, k-1\}$ and $J = \{k+1, \ldots, N\}$. The dissimilarity matrix $\mathbf{D}'$ satisfies*

$$d'_{min}(I, k) \leq d'_{min}(I, J). \tag{15}$$

Notice that this Lemma essentially shows that the property of $\mathbf{D}^*$ proven in Lemma II.1 applies to $\mathbf{D}'$.

Proof: Consider the MST edges that must be cut in order to produce the MST subtree I and the MST subtrees in J. By definition of VAT and Prim's algorithm the weight of the MST edge that connects k to I is less than or equal to the weights of the MST edges that connect I to the subtrees in J (if this was not true, then the vertices would be differently ordered by VAT). All MST paths from I to J must pass through the MST edges that are cut, thus the lower-bound on $d'_{min}(I, J)$ is the weight of these edges. The lower-bound on $d'_{min}(I, k)$ is the weight of the MST edge that connects k to I; hence, $d'_{min}(I, k) \leq d'_{min}(I, J)$.

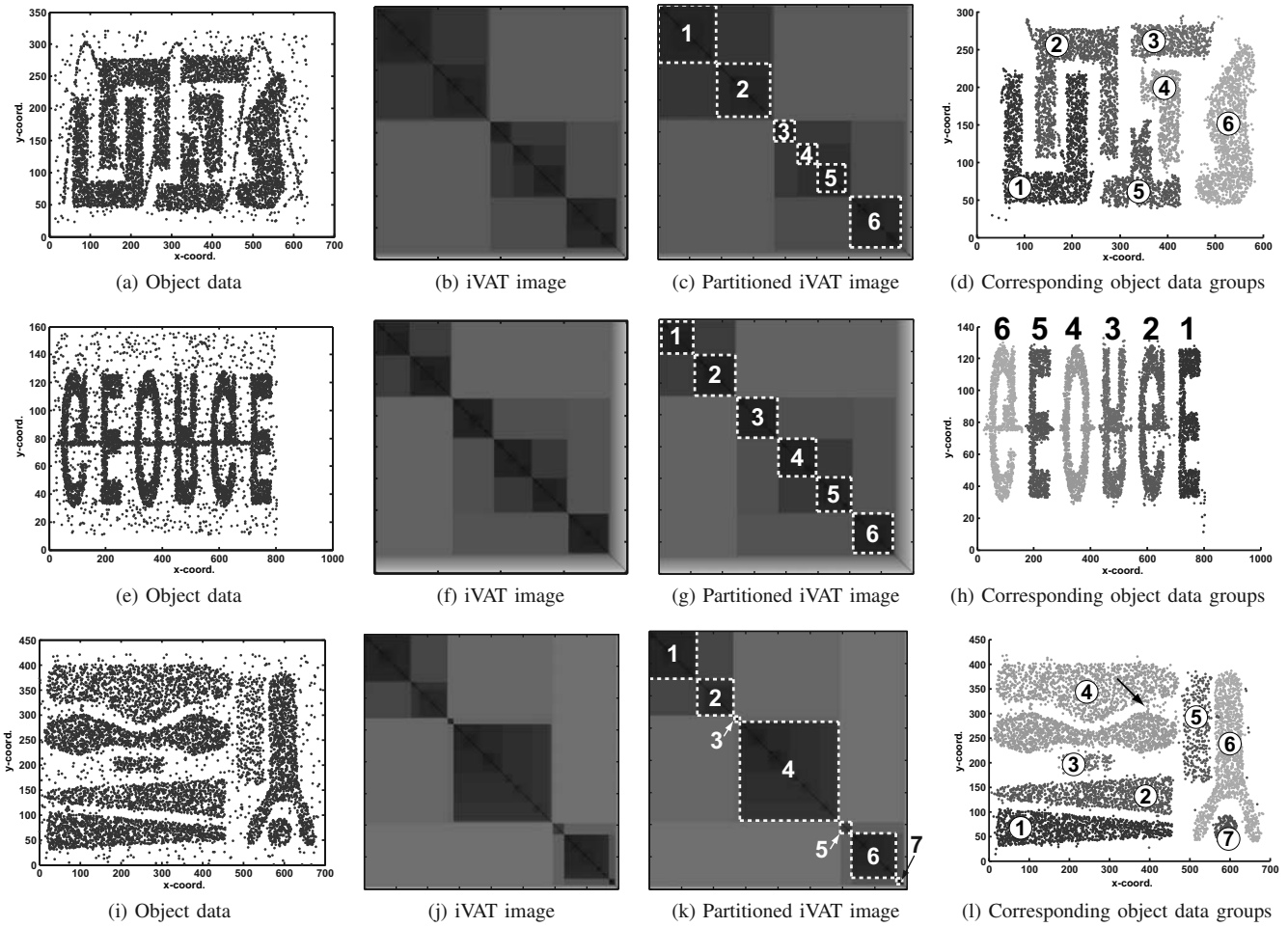
Note that a special case to consider is where there is one MST subtree in J and this is connected to I through an MST edge to vertex k. In this special case, it is easy to see that all MST paths from I to J must pass through k and thus (15) is true. ■

Remark 5. There are many possible VAT-reorderings of $\mathbf{D}'$ because there are many *ties* in the path-based distances and any object could be chosen as the initial object. Lemmas II.4 and II.5 show that $\mathbf{D}'$, as calculated by the formulas in Lemmas II.2 and II.3, is already in one possible VAT reordering. And, arguably, this ordering of $\mathbf{D}'$ is the "best" reordering because it is also the VAT-reordering of the original dissimilarity matrix.

III. Numerical Examples

The data sets used as examples in this section were initially created by the Karypis Lab for evaluating the CHAMELEON clustering algorithm [33] and can be downloaded at `http://glaros.dtc.umn.edu/gkhome/cluto`. Each data set is composed of 8,000 two-dimensional objects; hence, they can be easily visualized using 2D data plots. The dissimilarity data were built with a Euclidean distance relation. Figures 2(a,e,i) show plots of the data sets.

As view (a) illustrates, the first data set is composed of 6 dense regions with "noise" objects throughout and a "sine wave"-like structure that connects all the groups. View (b) demonstrates iVAT on this data set. The iVAT image



(a) Object data (b) iVAT image (c) Partitioned iVAT image (d) Corresponding object data groups

(e) Object data (f) iVAT image (g) Partitioned iVAT image (h) Corresponding object data groups

(i) Object data (j) iVAT image (k) Partitioned iVAT image (l) Corresponding object data groups

Fig. 2: Three examples of iVAT images of dissimilarity data built by Euclidean distance of 2-dimensional object data. Groups shown in views (c,d,g,h,k,l) were picked and labeled by identifying corresponding dark blocks in the iVAT image.

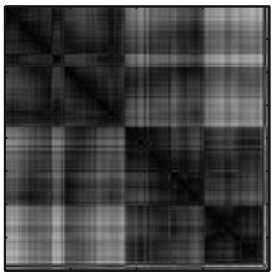
clearly shows 3 distinct dark blocks, with a substructure that indicates 6 clusters. We have (manually) outlined the 6 dark blocks in view (c) and the corresponding plot of the 6 groups of objects in view (d).

View (e) of Fig. 2 shows a data set composed of 6 dense regions connected by a line-like structure, with noise throughout. The iVAT image, in view (f), clearly shows 6 dark blocks. Views (g) and (h) demonstrate the these 6 dark blocks correspond to the 6 letters in 'GEORGE' (for George Karypis).

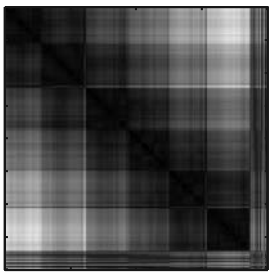
Finally, view (i) of Fig. 2 illustrates a data set that is composed of groups of differing sizes and density. There are four large dense regions on the left and one smaller dense region on the middle-left. A less dense region is shown on the right, next to an upside-'Y' structure that encapsulates a dense circular region at the bottom-right. Overall, we believe that these object data comprise 8 clusters. iVAT is partially successful for this data set as it only is able to show 7 of the 8 clusters. Views (k,l) shows the correspondence between the 7 dark blocks in the iVAT image and groups in the object data. As these images illustrate, iVAT is unable to delineate the cluster in the upper-left (indexed by the number '4'). This is because these two groups are joined by a line of close-proximity objects, shown by the arrow. The local density of this "bridge" of objects is similar to that of the two clusters they connect, denoted '4'. It would seem that iVAT would have similar troubles with the 'GEORGE' data set because of the horizontal line across the middle. However, the density of the objects in this horizontal line is less than the density of the objects in the six letter-shaped clusters. Thus, iVAT is able to distinguish these clusters accurately. Like any clustering algorithm, iVAT will "fail" on certain types of data. However, we believe that the examples in Fig. 2 show that iVAT is effective at showing the cluster tendency of data sets that are composed of odd-shaped groups obscured by both noise (or outliers) and *spoofing* objects, such as the sine-wave in view (a) and horizontal line in view (e).

For a final comparison, the VAT images of the data sets in Fig. 2 are shown in Fig. 3. While the VAT image in view (b) of the 'GEORGE' data set might suggest 6 clusters, the other VAT images do not show any clear cluster tendency.

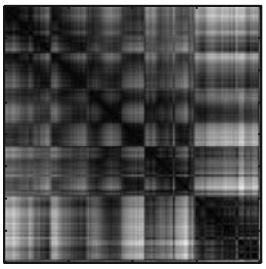
Recall that the VAT images are in the same matrix order as the iVAT images. Furthermore, the iVAT images show that the objects are ordered correctly; that is, the objects in



(a) VAT image of Fig. 2(a)



(b) VAT image of Fig. 2(e)



(c) VAT image of Fig. 2(i)

Fig. 3: VAT images of examples shown in Fig. 2

each group are ordered next to each other. Thus, any matrix reordering method that shows the dissimilarity data directly is going to suffer from the same problem that VAT does. For example, one could use any of the graph-based methods by thresholding $\mathbf{D}$ into a binary connection matrix $\mathbf{C}$. Then, the reordered $\mathbf{C}$ and a corresponding reordered $\mathbf{D}$ could be viewed.

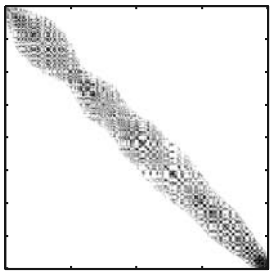
Figure 4 shows the results of using the RCM algorithm to reorder the data sets in Fig. 2. First, we thresholded $\mathbf{D}$ to produce a connection matrix $\mathbf{C}_\alpha$ by

$$(C_\alpha)_{ij} = \begin{cases} 1 & D_{ij} < \alpha \\ 0 & \text{else} \end{cases}, \forall i, j \tag{16}$$

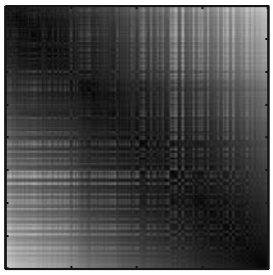
where α is a threshold. For the examples shown here, we used a threshold of $\alpha = 50$, which we empirically determined as the threshold that produced the most pleasing results. In practice, one could choose a few different thresholds and look at all the results (albeit, at a computational cost) or choose the threshold by some statistical method. Views (c,d) of Fig. 4 demonstrates that, like VAT, the RCM algorithm is able to accurately show the 6 clusters in the 'GEORGE' data set. However, the other RCM images show no cluster structure at all—arguably, the results in views (a,b,e,f) are even worse than the VAT results. Clearly, iVAT produces much more useful visualizations of the cluster tendency of these data sets.

A. Run-Time Comparison

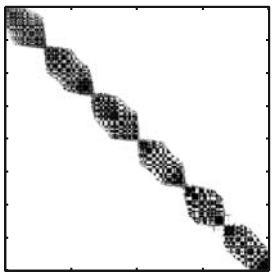
Figure 5 shows the normalized run-time of the RCM and iVAT matrix reordering methods. The run-time values were normalized to the run-time of the RCM method on a 500×500 matrix. Each value in the plot is the mean of 5 runs on a dissimilarity matrix computed by a Euclidean relation on



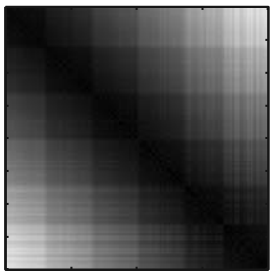
(a) RCM connection image of Fig. 2(a)



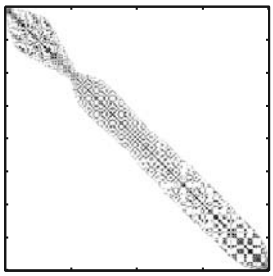
(b) RCM dissimilarity image of Fig. 2(a)



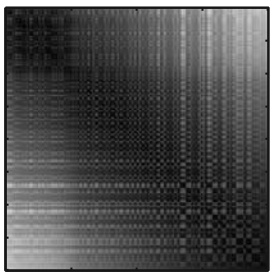
(c) RCM connection image of Fig. 2(e)



(d) RCM dissimilarity image of Fig. 2(e)



(e) RCM connection image of Fig. 2(i)



(f) RCM dissimilarity image of Fig. 2(i)

Fig. 4: RCM images of examples shown in Fig. 2

objects randomly placed in the 2-dimensional unit square. The evaluations were performed on a machine with an Intel Core 2 Duo, running at 2.2 GHz, with 4 GB of memory using the MATLAB [34] technical computer software.

The iVAT alorithm has, on average, a constant factor of 2 greater run-time than the RCM method. Our empirical observations of the results of these algorithms clearly show that the iVAT method is preferred; hence, a constant factor difference in run-time is acceptable.

B. iVAT Compared to RCM on Path-Based Distance

For our last experiment, we compared iVAT to the RCM algorithm operated on the path-based distance in (1). To our knowledge, our iVAT method is the most efficient instantiation of the all-pairs, shortest-path problem for the specific path-based distance in (1). Hence, we use our iVAT method to compute the path-based dissimilarity matrix; then we use RCM to produce the reordered matrix for visualization.

Figures 6-8 shows the results of the RCM visualization for multiple values of the threshold α, in (16), with $\mathbf{D}'$ as the input dissimilarity matrix.

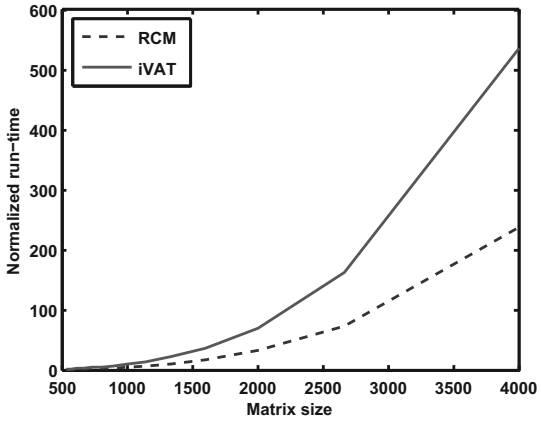
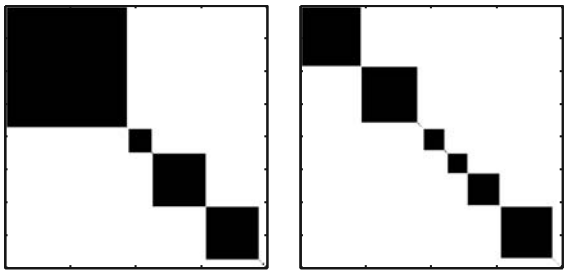


Fig. 5: Run-time comparison of RCM and iVAT matrix reordering methods — values are normalized to RCM run-time on a 500×500 matrix.

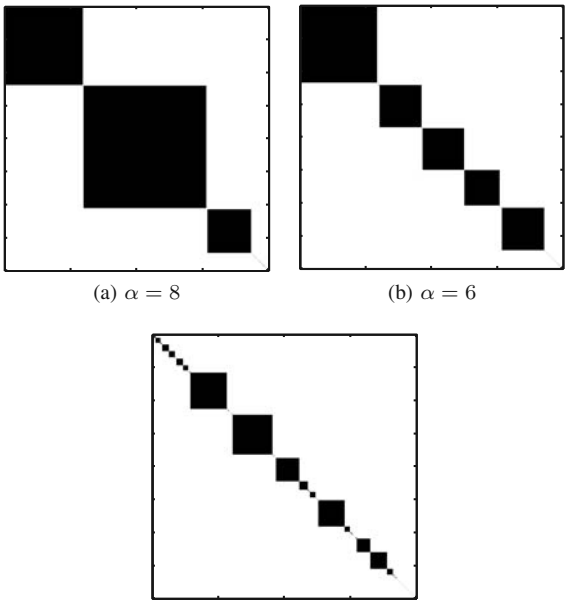


(a) $\alpha = 8$ (b) $\alpha = 7$

(c) $\alpha = 6$

Fig. 6: The results of the RCM algorithm on the path-based dissimilarity $\mathbf{D}'$ for the data shown in Fig. 2(a). α is the threshold used to create the binary connection matrix. The mean value of $\mathbf{D}'$ for this data set is 13.

The RCM visualization of the first data set—plotted in Fig. 2(i)—shows that the effectiveness of the RCM visualization in showing the correct number of clusters is very sensitive to the threshold α chosen. View (a) shows a threshold of $\alpha = 8$, view (b) of $\alpha = 7$, and view (c) of $\alpha = 6$. The mean value of the input dissimilarity matrix $\mathbf{D}'$ for this data set is 13. The visualization in view (a) shows 4 clusters, while view (b) shows 6 clusters and view (c) shows, arguably, 8 clusters. If we look back at the object data in Fig. 2(i), we see that there are 6 clusters in this data set. Hence, a threshold of $\alpha = 7$ is necessary for RCM, operated on $\mathbf{D}'$, to accurately show the tendency of this data set. A



(a) $\alpha = 8$ (b) $\alpha = 6$

(c) $\alpha = 4$

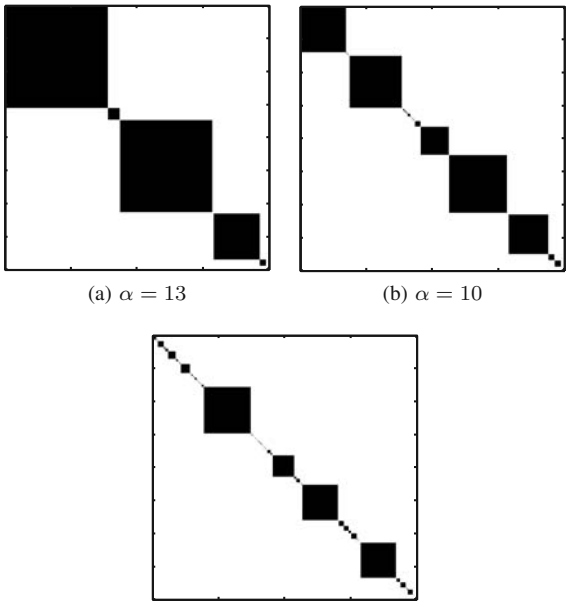
Fig. 7: The results of the RCM algorithm on the path-based dissimilarity $\mathbf{D}'$ for the data shown in Fig. 2(e). α is the threshold used to create the binary connection matrix. The mean value of $\mathbf{D}'$ for this data set is 8.

threshold of $\alpha = 8$ is too high and a threshold of $\alpha = 6$ is too low. We argue that this evidence alone is enough for us to tout the relatively good performance of iVAT. First, in order to produce the RCM images in Fig. 6, we first had to transform the distance matrix to the path-based distance matrix $\mathbf{D}'$. Because iVAT is the most efficient way to do this, we already have the iVAT image before the RCM image is even computed. Finally, RCM is shown in this example to be very sensitive to the threshold chosen. Hence, we recommend using the iVAT image directly to judge the cluster tendency of a data set. For the sake of completeness, we now examine this same procedure on the other two data sets shown in Fig. 2.

Figures 7 and 8 show the results of using RCM on the transformed distance matrix $\mathbf{D}'$ for different thresholds α for the data sets in Fig. 2, views (e) and (i), respectively. As was seen in the previous example, the effectiveness of RCM used on $\mathbf{D}'$ is very sensitive to the threshold α chosen. It only takes a small change in α to go from a visualization that shows too many clusters to a visualization that shows too few. This provides further evidence that our iVAT algorithm is more effective than RCM, even if RCM is used on the transformed distance matrix $\mathbf{D}'$.

IV. Discussion and Conclusion

The iVAT algorithm proposed in [10] was shown to improve the quality of VAT images, which ideally improves the interpretability of clustering tendency. The efficient formulation we propose in this paper significantly reduces the computational complexity of the iVAT algorithm; our formu-



(a) $\alpha = 13$ (b) $\alpha = 10$

(c) $\alpha = 8$

Fig. 8: The results of the RCM algorithm on the path-based dissimilarity $\mathbf{D}'$ for the data shown in Fig. 2(i). α is the threshold used to create the binary connection matrix. The mean value of $\mathbf{D}'$ for this data set is 14.

lation has a complexity of $O(N^2)$, compared to $O(N^3)$ for the formulation presented in [10]. Moreover, our algorithm produces both the VAT and iVAT images, which is not the case for the original iVAT method.

In every test of VAT versus iVAT we have either seen or run, the iVAT image was at least as good (visually) as the VAT image. Our conjecture is that iVAT images will always be equal to or superior to VAT images, but to date we have not discovered a way to prove this. Because this is a subjective evaluation, it may not be provable at all. Nonetheless, until a counterexample is discovered, we believe that with our formulation, there is no reason not to default to iVAT in every instance.

Additionally, we have inserted our efficient formulation of iVAT Eq.(1) into the *improved* co-VAT (co-iVAT) algorithm for visual assessment of clustering tendency in rectangular dissimilarity data [35, 36]. The co-iVAT algorithm seemed to improve the quality of co-VAT images much the same as the iVAT images do for VAT.

We are currently working on extending the iVAT algorithm to the scalable versions of VAT and co-VAT, *scalable VAT* (sVAT) [37], bigVAT [38], and *scalable co-VAT* (scoVAT) [39]. These algorithms work with very-large data, or data that is unloadable on standard computers. Perhaps counter-intuitively, the importantance of complexity reduction—from $O(N^3)$ to $O(N^2)$—for this particular application shrinks as N increases, because it become impossible to display very-large dissimilarity images due to resolution limitations imposed by current graphics hardware. Algorithms, such as bigVAT and sVAT, like VAT, are also $O(N^2)$ on the data they process. Their success depends not on improving complexity reduction, but rather, on processing a manageable *sample* of the input data matrix for accurate display purposes. Like VAT, however, the performance of these algorithms will suffer for "tough" data sets (see Fig. 3). Indeed, the efficient iVAT method developed in this paper will apply directly to bigVAT and sVAT, resulting in an improved visualization of very-large data with a complexity of $O(N^2)$.

We are also examining iVAT images as inputs to the *Clustering in Ordered Dissimilarity Data* (CLODD) [40], *Dark Block Extraction* (DBE) [41], and *Cluster Count Extraction* (CCE) [42] algorithms.

REFERENCES

[1] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 4th ed. San Diego, CA: Academic Press, 2009.

[2] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. Wiley-Interscience, October 2000.

[3] J. Hartigan, *Clustering Algorithms*. New York: Wiley, 1975.

[4] R. Xu and D. Wunsch II, *Clustering*. Psicataway, NJ: IEEE Press, 2009.

[5] A. Jain, M. Murty, and P. Flynn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, September 1999.

[6] A. Jain and R. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

[7] D. Jiang, C. Tang, and A. Zhang, "Cluster analysis for gene expression data: a survey," *IEEE Trans. Knowledge Engr.*, vol. 16, no. 11, pp. 1370–1386, Nov. 2004.

[8] A. Jain, "Data clustering: 50 years beyond k-means," in *Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science, W. Daelemans, B. Goethals, and K. Morik, Eds. Springer Berlin / Heidelberg, 2008, vol. 5211, pp. 3–4.

[9] J. Bezdek and R. Hathaway, "VAT: A tool for visual assessment of (cluster) tendency," in *Proc. IJCNN*, Honolulu, HI, 2002, pp. 2225–30.

[10] L. Wang, T. Nguyen, J. Bezdek, C. Leckie, and K. Ramamohanarao, "iVAT and aVAT: enhanced visual analysis for cluster tendency assessment," in *Proc. PAKDD*, Hyderabad, India, Jun. 2010.

[11] W. Petrie, "Sequences in prehistoric remains," *J. Anthropological Inst. Great Britain and Ireland*, vol. 29, pp. 295–301, 1899.

[12] J. Czekanowski, "Zur differentialdiagnose der neandertalgruppe," *Korrespondenzblatt der Deutschen Gesellschaft fr Anthropologie, Ethnologie und Urgeschichte*, vol. 40, pp. 44–47, 1909.

[13] L. Wilkinson and M. Friendly, "The history of the cluster heat map," *The American Statistician*, vol. 63, no. 2, pp. 179–184, 2009.

[14] R. Tryon, *Cluster Analysis*. Ann Arbor, MI: Edwards Bros., 1939.

[15] R. Cattell, "A note on correlation clusters and cluster

search methods," *Psychometrika*, vol. 9, pp. 169–184, 1944.

[16] P. Sneath, "A computer approach to numerical taxonomy," *J. Gen. Microbiol.*, vol. 17, pp. 201–226, 1957.

[17] G. Floodgate and P. Hayes, "The Adansonian taxonomy of some yellow pigmented marine bacteria," *J. Gen. Microbiol.*, vol. 30, pp. 237–244, 1963.

[18] R. Ling, "A computer generated aid for cluster analysis," *Communications of the ACM*, vol. 16, no. 6, pp. 355–361, 1973.

[19] D. Johnson and D. Wichern, *Applied Multivariate Statistical Analysis*, 6th ed. Englewood Cliffs, NJ: Prentice Hall, 2007.

[20] T. Tran-Luu, "Mathematical concepts and novel heuristic methods for data clustering and visualization," Ph.D. dissertation, U. of Maryland, College Park, MD, 1996.

[21] M. Girolami, "Mercer kernel-based clustering in feature space," *IEEE Trans. Neural Networks*, vol. 13, pp. 780–784, 2002.

[22] D. Zhang and S. Chen, "Clustering incomplete data using kernel-based fuzzy c-means algorithm," *Neural Processing Letters*, vol. 18, pp. 155–162, 2003.

[23] D. West, *Introduction to Graph Theory*, 2nd ed. Prentice Hall, Inc., 2001.

[24] C. Mueller, B. Martin, and A. Lumsdaine, "A comparison of vertex ordering algorithms for large graph visualization," in *Proc. Int. Asia-Pacific Symp. Visualization*, Sydney, Australia, Feb. 2007, pp. 141–148.

[25] A. George and J. Liu, *Computer Solution of Large Sparse Positive Definite Systems*. Prentice-Hall, 1981.

[26] I. King, "An automatic reordering scheme for simultaneous equations derived from network analysis," *Int. J. Numerical Methods in Engineering*, vol. 2, p. 523533, 1970.

[27] S. Sloan, "An algorithm for profile and wavefront reduction of sparse matrices," *Int. J. Numerical Methods in Engineering*, vol. 23, pp. 239–251, 1986.

[28] R. Prim, "Shortest connection networks and some generalisations," *Bell System Tech. J.*, vol. 36, pp. 1389–1401, 1957.

[29] B. Fisher, T. Zoller, and J. Buhmann, "Path based pairwise data clustering with application to texture segmentation," *Energy Minimization Methods in Computer Vision and Pattern Recognition*, vol. 2134, pp. 235–250, 2001.

[30] J. Tenenbaum, V. de Silva, and J. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.

[31] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA: MIT Press, 2009.

[32] T. Havens, J. Bezdek, J. Keller, M. Popescu, and J. Huband, "Is VAT really single linkage in disguise?" *Ann. Math. Artif. Intell.*, vol. 55, no. 3-4, pp. 237–251, 2009.

[33] G. Karypis, E. Han, and V. Kumar, "CHAMELEON: a hierarchical clustering algorithm using dynamic modeling," *IEEE Computer*, vol. 32.

[34] *Using MATLAB*, The Mathworks, Natick, MA, November 2000.

[35] T. Havens, J. Bezdek, and J. Keller, "A new implementation of the co-VAT algorithm for visual assessment of clusters in rectangular relational data," in *Artficial Intelligence and Soft Computing, Part I*. Berlin: Springer, Apr. 2010, pp. 363–371.

[36] J. Bezdek, R. Hathaway, and J. Huband, "Visual assessment of clustering tendency for rectangular dissimilarity matrices," *IEEE Trans. Fuzzy Systems*, vol. 15, no. 5, pp. 890–903, October 2007.

[37] R. Hathaway, J. Bezdek, and J. Huband, "Scalable visual asseessment of cluster tendency for large data sets," *Pattern Recognition*, vol. 39, no. 7, pp. 1315–1324, July 2006.

[38] J. Huband, J. Bezdek, and R. Hathaway, "bigVAT: visual assessment of cluster tendency for large data sets," *Pattern Recognition*, vol. 38, no. 11, pp. 1875–1886, November 2005.

[39] L. Park, J. Bezdek, and C. Leckie, "Visualization of clusters in very large rectangular dissimilarity data," in *Proc. 4th Int. Conf. Autonomous Robots and Agents*, G. S. Gupta and S. Mukhopadhyay, Eds., Feb. 2009, pp. 251–256.

[40] T. Havens, J. Bezdek, J. Keller, and M. Popescu, "Clustering in ordered dissimilarity data," *Int. J. Intell. Syst.*, vol. 24, no. 5, pp. 504–528, May 2009.

[41] L. Wang, C. Leckie, K. Rao, and J. Bezdek, "Automatically determining the number of clusters from unlabeled data sets," *IEEE Trans. Knowledge Engr.*, vol. 21, no. 3, pp. 335–350, March 2009.

[42] I. Sledge, T. Havens, J. Huband, J. Bezdek, and J. Keller, "Finding the number of clusters in ordered dissimilarities," *Soft Computing*, vol. 13, no. 12, pp. 1125–1142, October 2009.

Tim Havens is an NSF / CRA Computing Innovation Fellow in the Department of Computer Science and Engineering at Michigan State University. He received an M.S. in electrical engineering from Michigan Tech University in 2000 and a Ph.D. in electrical and computer engineering from the University of Missouri in 2010. Prior to his Ph.D. work, he was employed at MIT Lincoln Laboratory where he specialized in the simulation and modeling of directed energy and global positioning systems. He received a best paper award from the Midwest Nursing Research Society (2009). He is a senior member of the IEEE and an accomplished jazz bassist.

His research interests include machine learning, clustering, fuzzy logic, informatics, and pattern recognition.

Jim Bezdek received the Ph.D. in Applied Mathematics from Cornell University in 1973. Jim is past president of NAFIPS, IFSA and the IEEE CIS; founding editor of the International Journal of Approximate Reasoning and the IEEE Transactions on Fuzzy Systems; life fellow of the IEEE and IFSA; and a recipient of the IEEE 3rd Millennium, IEEE CIS Fuzzy Systems Pioneer, and IEEE (TFA) Rosenblatt medals.

Jim's interests include woodworking, cluster validity, motorcycles, pattern recognition, cigars, clustering in very large data, fishing, visual methods for clustering, blues music, wireless sensor networks, poker, and co-clustering in rectangular relational data. Jim retired in 2007 and will be coming to a university near you soon (especially if there is fishing nearby).