Scatter/Gather Clustering: Flexibly Incorporating User Feedback to Steer Clustering Results

M. Shahriar Hossain, *Member, IEEE*, Praveen Kumar Reddy Ojili, Cindy Grimm, Rolf Müller, Layne T. Watson, *Fellow, IEEE*, Naren Ramakrishnan

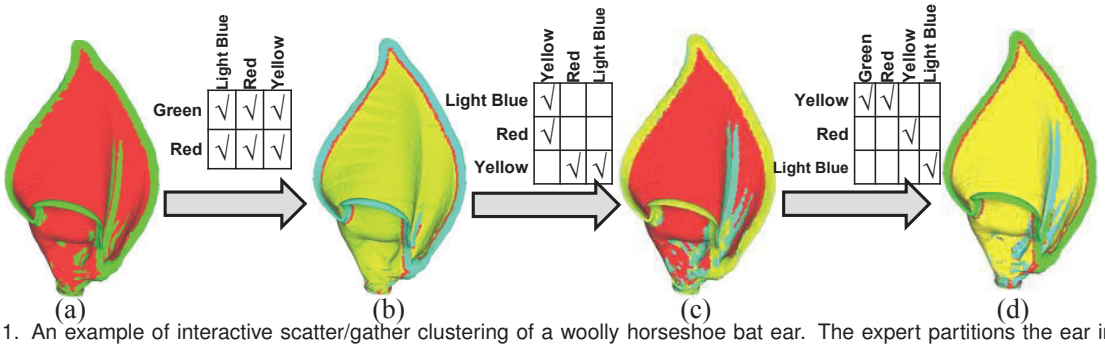


Figure 1. An example of interactive scatter/gather clustering of a woolly horseshoe bat ear. The expert partitions the ear into four clusters beginning from a setting of two clusters. *(a) to (b)*—The expert supplies a 2×3 constraint table to generate three clusters from two, and the vertical ridge is lost in the result; *(b) to (c)*—the expert supplies constraints in a 3×3 table to retrieve the vertical ridge; *(c) to (d)*—the expert provides constraints in a 3×4 matrix to scatter the border into two layers but to keep the rest of the clusters the same.

Abstract— Significant effort has been devoted to designing clustering algorithms that are responsive to user feedback or that incorporate prior domain knowledge in the form of constraints. However, users desire more expressive forms of interaction to influence clustering outcomes. In our experiences working with diverse application scientists, we have identified an interaction style scatter/gather clustering that helps users iteratively restructure clustering results to meet their expectations. As the names indicate, *scatter* and *gather* are dual primitives that describe whether clusters in a current segmentation should be broken up further or, alternatively, brought back together. By combining scatter and gather operations in a single step, we support very expressive dynamic restructurings of data. Scatter/gather clustering is implemented using a nonlinear optimization framework that achieves both locality of clusters and satisfaction of user-supplied constraints. We illustrate the use of our scatter/gather clustering approach in a visual analytic application to study baffle shapes in the bat biosonar (ears and nose) system. We demonstrate how domain experts are adept at supplying scatter/gather constraints, and how our framework incorporates these constraints effectively without requiring numerous instance-level constraints.

Index Terms—Scatter/gather clustering, alternative clustering, constrained clustering.

1 Introduction

Clustering is a classical technique for data analysis and has become increasingly repurposed for new uses, with the advent of novel applications in bioinformatics [45,56,63], intelligence analysis [41,51], and web modeling [1, 43]. Of recent interest has been the ability to impart prior domain knowledge in the form of constraints [22,23,61,62], clustering nonhomogeneous datasets [32], or providing expressive forms of user feedback [4, 33, 34].

We were motivated by the iterative process by which users inspect clustering results, rerun clustering with different settings (e.g., changing the number of clusters), and assess the new results. In particular, our desire was to provide a very natural interface for users by which they can critique results and, at the same time, operationalize their feedback into an effective mechanism to recluster the results. Our thesis is that 'a little domain knowledge goes a long way', and enabling the user in the loop to supply feedback can be significantly more effective than trying to design a clever clustering algorithm.

We introduce a novel visual analytic approach—*scatter/gather clustering*—that enables users to iteratively restructure clustering results to meet their expectations. As the names indicate, *scatter* and *gather* are dual primitives that describe whether clusters in the current segmentation should be broken up further or, alternatively, brought back together. We will demonstrate how, by mixing scatter and gather operations in a sequence of interactions (Fig. 1), users can very quickly arrive at a segmentation of choice.

Our contributions are:

1. A new interaction style to steer clustering results and, correspondingly, an underlying mathematical optimization framework to support such restructurings. Further, our framework subsumes previously introduced clustering variations such as *alternative clustering* [19–21, 52].
2. A systematic approach to compose scatter/gather operations so that our framework can be applied to the results of *any* clustering algorithm.

- *M. S. Hossain is with the Department of Computer Science and the Discovery Analytics Center, Virginia Tech, e-mail: msh@vt.edu.*
- *P. K. R. Ojili is with the Department of Mechanical Engineering, Virginia Tech, e-mail: opkreddy@vt.edu.*
- *C. Grimm is with the Department of Computer Science and Engineering, Washington University, St. Louis, e-mail: cmg@cse.wustl.edu.*
- *R. Müller is with the Department of Mechanical Engineering, Virginia Tech, e-mail: rolf.mueller@vt.edu.*
- *L. T. Watson is with the Department of Computer Science, the Department of Mathematics, and the Discovery Analytics Center, Virginia Tech, e-mail: ltwatson@computer.org.*
- *N. Ramakrishnan is with the Department of Computer Science and the Discovery Analytics Center, Virginia Tech, e-mail: naren@vt.edu.*

Manuscript received 31 March 2012; accepted 1 August 2012; posted online 14 October 2012; mailed on 5 October 2012.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

3. A novel visual analytics application to studying structural patterns of baffle shapes in the bat biosonar system that systematizes how acoustics and vibration experts supply domain knowledge.

2 Related Work

We survey related work under different categories.

Scatter/Gather Browsing and Interaction Mechanisms: The phrase 'scatter gather' was actually coined in reference to a document browsing/retrieval strategy [17, 18]. To communicate the structure of a document collection, the idea here is to first scatter (cluster) the documents into groups, gather (collect) a subset of results, scatter them again, and so on. We were motivated by the underlying iterative strategy but use the terms *scatter* and *gather* with different interpretations here. The scatter/gather approach of [17, 18] is meant to narrow down to a single or few data points (documents) from a collection of points, whereas our scatter/gather strategy retains all data points at all times and is focused on iteratively reorganizing them into clusters. Thus the semantics of the scatter and gather operations are fundamentally different. In particular, we allow scattering and gathering to take place together in a single interaction and that there can be complex dependencies between scattering and gathering. Further, our approach works for any dataset rather than just document collections (as we will show in this paper). Nevertheless, the work of [17, 18] was pioneering in its embrace of interaction as a way to retrieve better quality results and the use of clustering as a modality for doing so. Alonso and Talbot [4] propose an extension with their 'exposed hierarchical tree view' that is incrementally built as the user explores the collection. Here, the root node represents the entire document collection and other nodes represent clusters of documents produced by the scatter/gather operations. Kanada [37] introduces an axis-based organization method for search results. For example, a search result with a specific query could be ordered by axes like time, size, area, and other units. All these methods, as mentioned earlier, are focused on document collections and involve alternating applications of scatter and gather operations. A recent paper [30] describes the idea of scatter-gather as a technique to browse trajectories discovered from surveillance videos. Here we employ scatter and gather operations as primitives to restructure clusters in a more expressive manner.

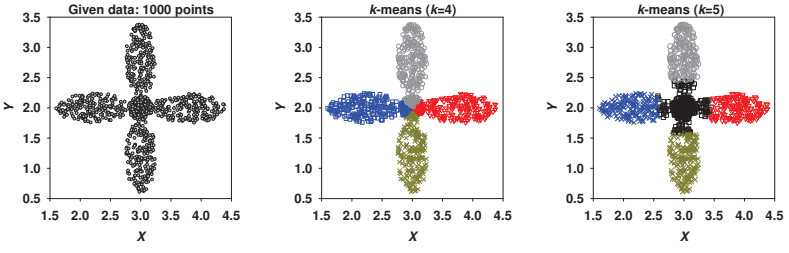
Visual Analytic Frameworks for Clustering: The necessity of supervision for clustering has motivated several works [3,6–8,24,48,55]. Schreck et al. [55] describe a visual analytic framework to effectively combine automatic data analysis with expert supervision. This framework has been applied on a trajectory clustering problem which demonstrates its potential of combining machine and user-directed processing in producing appropriate cluster results. Jeong et al. [35] combine visualization with clustering to create tools for visual analysis of gene expression data. G. Andrienko & N. Andrienko [5] and Guo et al. [28] incorporate space and time into clustering to support visual interactions with data. There are optimization techniques to group dimensions of data [7] as well as dimensionality reduction [3, 13] tools that can preserve clustering quality. Nam et al. [48] and Chen and Ling [11] describe interactive clustering systems in which users can interact with the data objects after they are clustered.

Constrained Clustering: In the machine learning domain, constrained clustering refers to the idea of incorporating user supervision into a clustering algorithm. Instance-level pairwise constraints can appear in two forms: must-link and must-not-link [60, 62]. Constrained clustering algorithms proposed in [23,60,62] attempt to find a solution to satisfy all the must-link and must-not-link constraints. It is sometimes cumbersome for the user to provide such instance-level feedback because there can be numerous combinations of the must-link and must-not-link constraints. In our work, the user provides scatter/gather constraints *at the cluster level* (rather than at the instance-level) and hence the constraints are very small in number, easy to provide, and intuitive to understand.

Finding Multiple Clusterings: The idea of finding more clusterings than a single one has been studied through various mechanisms and also in various guises, including subspace clustering [2,12], nonredundant clustering/views [16, 25, 49], associative clustering [38, 58], meta clustering [10, 64], and consensus clustering [40, 45, 59]. A key distinguishing feature of our work is the ability to interactively provide feedback to obtain variations in clusterings. As we will show below, our objective functions for scatter/gather clustering employ a simple contingency table framework. While contingency tables have been employed elsewhere [9, 57], they have been used primarily as criteria to evaluate clusterings, not to specify requirements on clusterings. The few works [26, 27, 47] that do use contingency tables to formulate objective criteria use them in the context of a specific algorithm such as co-clustering or block clustering, whereas we use them to specify scatter and gather operations. Our work can also be viewed as a form of relational clustering [32] because we use (two) homogeneous copies of the data to model the scatter/gather property of two clusterings. However, the locality of clusterings in their respective data spaces is also incorporated into the objective function without any explicit trade-off between locality and the 'scatter/gather-ness' property.

3 Scatter/Gather Analytic Approach

Before we introduce our framework and the underlying mathematical machinery, it is helpful to consider a motivating example.



(a) Original data (b) k-means (k=4) (c) k-means (k=5)

Figure 2. Clustering the flower dataset. (a) The dataset has 1000 2D points arranged in the form of a flower. (b) Result of k-means clustering with k=4. (c) k-means clustering with k=5. Points from the stalk spill over into the petals.

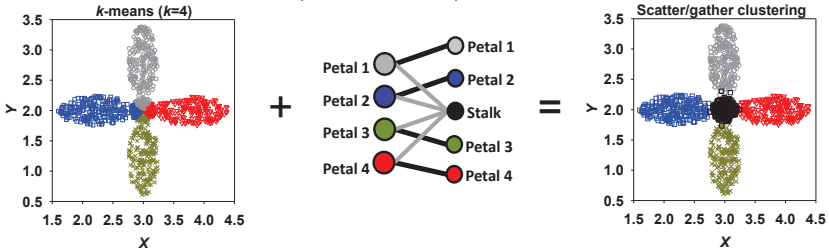


Figure 3. Clustering the flower dataset with user provided input: Scatter/gather constraints when imposed over a clustering with four clusters yields five clusters with well-separated petals and center with the stalk, unlike Fig. 2(c).

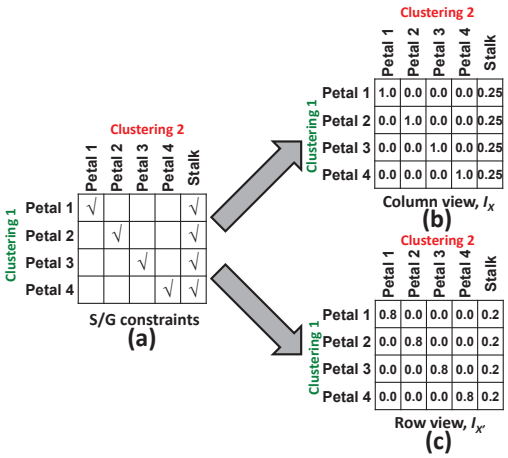


Figure 4. (a) A tabular representation of the scatter/gather constraints in the middle of Fig. 3, (b) Matrix for column-wise distribution, (c) Matrix for row-wise distribution.

3.1 Motivating Example

To illustrate the idea of scatter/gather clustering, we use a synthetic dataset composed of 1000 two-dimensional points (see Fig. 2(a)). The dataset is composed of four petals and a stalk each containing 200 points. When the user applies simple k-means clustering, with a setting of four clusters (i.e., $k = 4$), the flower is divided into four parts as shown in Fig. 2(b) where the petals are indeed in different clusters, but each of the petals also takes up one-fourth of the points from the stalk of the flower. When a setting of five clusters is used, the user obtains the clustering shown in Fig. 2(c). It is evident that the five clusters generated by k-means are not able to cleanly differentiate the stalk from the petals.

A conventional clustering algorithms like k-means does not take user expectation as an input to produce better clustering results. Even constrained clustering algorithms would require an inordinate number of inputs to clearly separate the stalk from the petals. In our proposed clustering framework, the user can provide an input to the algorithm regarding the expected outcome as shown in Fig. 3. The constraints shown in the middle of the figure should be read both from *left to right* and from *right to left*. Reading from left to right, we see that the user expects the four clusters to be broken down (scattered) into five clusters. Reading from right to left, we see that the stalk is expected to gather points from all current clusters, but there is a one-to-one correspondence between the desired petals to the original petals. Fig. 3 shows that the results of such a scatter/gather clustering provide well-separated petals and stalk, unlike the result provided by simple k-means with k=5 (as shown in Fig. 2(c)).

3.2 User Input

The scatter/gather framework thus requires an existing clustering of the data and a user-expected distribution of the clusters in the new clustering (as shown in Fig. 3). In this paper, we refer to the existing clustering as 'given clustering', 'clustering 1', or the 'first clustering'. We refer to the new clustering as 'clustering 2', or the 'second clustering'.

We enable the user to provide a set of scatter/gather constraints in the form of a matrix called an *S/G constraint table*. The matrix is essentially an encoding of the bipartite graph shown in Fig. 3. For our running example, the matrix is of size 4×5 as shown in Fig. 4(a), where each row indicates a petal of the given (k-means) clustering and a column indicates an expected cluster of the output of the scatter/gather clustering framework. The tick marks denote the scatter and gather operations desired. Note that each row of Fig. 4(a) has two tick marks and one of these tick marks is in the fifth column, which is the column for the expected stalk of the flower.

3.3 Formulating Probabilistic Contingency Tables

A cell of the S/G constraint table is meant to represent an expected (or an ideal case) probability that objects of a cluster in one clustering form part of a cluster in another clustering. We enable the user to supply a binary association matrix between two clusterings in the form of the S/G constraint table (e.g., Fig. 4(a)). This matrix is converted into two probability distributions, one defined along columns and one defined along rows. This results in two matrices, row view $I_{\mathcal{X}'}$ and column view $I_{\mathcal{X}}$ (see Fig. 4(b) and (c)). Although there are many ways to construct such distributions from the binary matrix, we perform simple row-wise and column-wise normalizations here. (More complex distributions can, of course, be incorporated based on user input.) Thus, in our example, although not explicitly mentioned by the user, we infer that 25% points of the stalk cluster should come from each of the petals of the first clustering. Conversely, these distributions capture the requirement that each of the petals of the first clustering should give up 20% of their points to form the stalk of the second clustering and that the other 80% of the points should go into one cluster of the second clustering.

3.4 Mathematical Framework

We now present the formalisms in our approach. Consider a dataset $\mathcal{X} = \{\mathbf{x}_s\}, s = 1, \ldots, n$, of (real-valued) l_x-dimensional vectors, i.e., $\mathbf{x}_s \in \mathbb{R}^{l_x}$. Because we desire two different sets of clusters from the scatter/gather clustering approach, we create $\mathcal{X}' = \mathcal{X}$ an exact replica of $\mathcal{X}$. Let $C_{(x)}$ and $C_{(x')}$ be the cluster indices, i.e., indicator random variables, corresponding to $\mathcal{X}$ and $\mathcal{X}'$ and let k and k' be the corresponding number of clusters. Thus, $C_{(x)}$ takes values in $\{1, \ldots, k\}$ and $C_{(x')}$ takes values in $\{1, \ldots, k'\}$. Among these two clusterings, the clustering of $\mathcal{X}$ is given and the clustering of $\mathcal{X}'$ is to be determined.

Let $\mathbf{m}_{i,\mathcal{X}}$ ($\mathbf{m}_{j,\mathcal{X}'}$) be the prototype vector for cluster i (j) in $\mathcal{X}$ ($\mathcal{X}'$). (These are precisely the quantities we wish to estimate/optimize, but in this section, assume they are given). Let $v_i^{(\mathbf{x}_s)}$ ($v_j^{(\mathbf{x}_t)}$) be the cluster membership indicator variables, i.e., the probability that data sample $\mathbf{x}_s$ ($\mathbf{x}_t$) is assigned to cluster i (j) in $\mathcal{X}$ ($\mathcal{X}'$). Thus, $\sum_{i=1}^{k} v_i^{(\mathbf{x}_s)} = \sum_{j=1}^{k'} v_j^{(\mathbf{x}_t)} = 1$. The traditional *hard* assignment is given by:

$$v_i^{(\mathbf{x}_s)} = \begin{cases} 1, & \text{if } ||\mathbf{x}_s - \mathbf{m}_{i,\mathcal{X}}|| \leq ||\mathbf{x}_s - \mathbf{m}_{i',\mathcal{X}}||, i' = 1, \ldots k, \\ 0, & \text{otherwise.} \end{cases}$$

(Likewise for $v_j^{(\mathbf{x}_t)}$.) Ideally, we would like a continuous function that tracks these hard assignments to a high degree of accuracy. A standard approach is to use a Gaussian kernel to smooth out the cluster assignment probabilities:

$$v_i^{(\mathbf{x}_s)} = \frac{\exp(-\frac{\rho}{D}||\mathbf{x}_s - \mathbf{m}_{i,\mathcal{X}}||^2)}{\sum_{i'=1}^{k} \exp(-\frac{\rho}{D}||\mathbf{x}_s - \mathbf{m}_{i',\mathcal{X}}||^2)}, \quad (1)$$

where

$$D = \max_{s,s'} ||\mathbf{x}_s - \mathbf{x}_{s'}||^2, 1 \leq s, s' \leq n.$$

An analogous equation holds for $v_j^{(\mathbf{x}_t)}$. The astute reader would notice that this is really the Gaussian kernel approximation with ρ/D being the width of the kernel. Notice that D is completely determined by the data but ρ is a user-settable parameter, and precisely what we can tune.

3.4.1 Preparing contingency tables

Contingency tables capture the relationships between entries in clusters across two clusterings (here the clusterings of $\mathcal{X}$ and $\mathcal{X}'$). To prepare a $k \times k'$ contingency table, we simply iterate over the implicit one-to-one relationships between $\mathcal{X}$ and $\mathcal{X}'$: We suitably increment the appropriate entry in the contingency table in a one-to-one relationship fashion:

$$w_{ij} = \sum_{m=1}^{n} v_i^{(\mathbf{x}_m)} v_j^{(\mathbf{x}_m)}, \quad (2)$$

We also define

$$w_{i.} = \sum_{j=1}^{k'} w_{ij}, \qquad w_{.j} = \sum_{i=1}^{k} w_{ij}$$

where $w_{i.}$ and $w_{.j}$ are the row-wise and column-wise counts of the cells of the contingency table, respectively.

We will find it useful to define the probability distribution $\alpha_i(j), i = 1, \ldots, k$ of the row-wise random variables and $\beta_j(i), j = 1, \ldots, k'$ of the column-wise random variables as

$$\alpha_i(j) = \frac{w_{ij}}{w_{i.}}, \qquad \beta_j(i) = \frac{w_{ij}}{w_{.j}}.$$

The row-wise distributions represent the conditional distributions of the clusters in $\mathcal{X}'$ given the clusters in $\mathcal{X}$; the column-wise distributions are also interpreted analogously.

3.4.2 Evaluating contingency tables

Now that we have a contingency table, we must evaluate it to see if it reflects disparateness of the two clusterings. Ideally, we expect that row-wise distribution α_i and column-wise distribution β_j of the contingency table would match with the row view $I_{\mathcal{X}'}$ and column view $I_{\mathcal{X}}$ of the expected contingency table generated from the S/G constraint table. Therefore for our objective criterion, we compare the row-wise and column-wise distributions from the contingency table entries to their corresponding row and column views of the expected contingency table generated from the user provided S/G constraint table. We use KL-divergences to define the objective function (lower values are better):

$$\begin{aligned}\mathcal{F} =& \frac{1}{k}\sum_{i=1}^{k} D_{KL}\Big(\alpha_i || I_{\mathcal{X}'}(i,:)\Big) + \frac{1}{k'}\sum_{j=1}^{k'} D_{KL}\Big(\beta_j || I_{\mathcal{X}}(:,j)\Big) \\ & - \frac{1}{n}\sum_{s=1}^{n} D_{KL}\Big(p\left(V^{(\mathbf{x}_s)}\right) || U(\frac{1}{k})\Big) \\ & - \frac{1}{n}\sum_{t=1}^{n} D_{KL}\Big(p\left(V^{(\mathbf{x}_t)}\right) || U(\frac{1}{k'})\Big), \end{aligned} \tag{3}$$

where $I_{\mathcal{X}'}(i,:)$ refers to the ith row of the row view and $I_{\mathcal{X}}(:,j)$ represents the jth column of the column view of the expected contingency table generated from the user provided S/G constraint table as described in Section 3.3. $p\left(V^{(\mathbf{x}_s)}\right)$ refers to the vector containing the cluster membership probabilities of the sth datapoint of $\mathcal{X}$ (likewise, $p\left(V^{(\mathbf{x}_t)}\right)$). U is the uniform distribution over k or k' clusters. (Note that the row-wise distributions take values over the columns and the column-wise distributions take values over the rows of the contingency table.)

3.4.3 Optimizing the objective function

Since scatter/gather clustering assumes that the clustering of $\mathcal{X}$ is given, we keep the mean prototypes of $\mathcal{X}$ fixed to k-means outcomes (or the current clustering) and vary the mean prototypes of $\mathcal{X}'$ during the optimization. As a result, we obtain a scatter/gather clustering in $\mathcal{X}'$ at the end of the optimization. We use an interior trust region based approach [15] for nonlinear minimization of our objective function $\mathcal{F}$.

3.5 Alternative Clustering: A Special Case of Scatter/Gather Clustering

When the numbers of clusters in the given and the expected clustering are the same and all the cells of the S/G constraint table are filled, a special case scenario named "alternative clustering" (popular in the KDD literature [19–21, 52]) is obtained. The goal of alternative clustering is to obtain two high-quality clusterings where the partitionings are as highly different from each other as possible. An example of an alternative clustering scenario with our running example dataset is shown in Fig. 5. Thus, scatter/gather clustering is a more expressive generalization of alternative clustering.

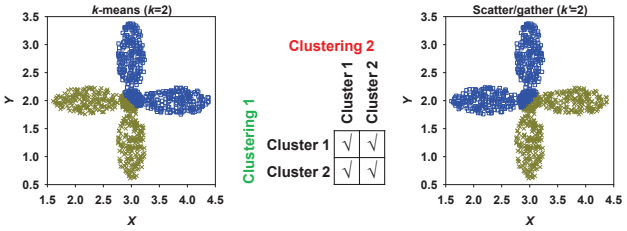


Figure 5. A special case of scatter/gather clustering where each cluster is constrained to share elements with all the clusters of the second clustering. (left) k-means clustering with k=2, (middle) S/G constraint table, (right) resulting scatter/gather clustering.

4 A Visual Analytic Framework to Study the Bat Biosonar System

We now illustrate a visual analytic framework based on scatter/gather clustering to study the bat biosonar system.

4.1 Background: Bat Biosonar System

In the course of evolution, bats have developed an ultrasonic sensory system with high performance, so called biosonar, that the majority of recent bat species rely on as an important far sense. Bat biosonar comprises four primary parts: signal generation (vocal folds), signal emission (mouth or nostril), signal reception (ear), and signal analysis (brain). Around 300 out of over thousand bat species presently known to science emit their ultrasonic biosonar pulses through the nostrils. All bats listen to incoming signals through their ears. The geometries of the external structures in the biosonar system of bats differ considerably between species. Since such differences could potentially be of great functional importance, they need to be considered when analyzing the function of bat biosonar.

The sound emission sites of bat species with nasal emission are surrounded by soft-tissue structures (noseleaves) with often intricate shape detail. The geometrical features of these structures could significantly influence the beamforming operations that are performed on the outgoing ultrasonic pulses. This hypothesis is corroborated by experimental case studies. Bats rely on their biosonar as a far sense to support navigation and the search for food [54] in their habitats. The biosonar systems of bats have undergone an extensive adaptive evolution to match different ecological niches [36]. Bats obtain sensory information through active sonar, i.e., the analysis of echoes to self-emitted pulses, as well as passive sonar, i.e., the analysis of sounds from foreign sources [53]. Active and passive sonar contribute important sensory information for the acquisition of food in diets as diverse as arthropods, vertebrates, nectar and pollen, fruit, and blood.

The pinnae of bats act as baffles that diffract the incoming ultrasonic waves. Hence, the shapes of the pinnae are in a position to play a key role in determining the distribution of the ear's sensitivity over direction and frequency [46, 50]. The pinnae can hence be seen as beamforming devices operating in the physical (diffraction) domain. Their function could inspire beamforming mechanisms as well as strategies for engineering applications.

There is no visual analytic tool to study the local shapes of the bat biosonar systems. In this work, we provide a scatter/gather clustering framework to help experts study the biosonar systems. Tools for deriving estimates of the acoustic functions from the shapes of the ears and noseleaves are readily available and have been used in several case studies already [44]. In the present work, we concentrate on the "shape" aspect.

Ma and Müller [42] have demonstrated that an eigensystem based approach called eigenears is good at capturing overall shape properties. However, interpreting the eigenears for local shape features is a difficult task. Nevertheless, these local shape features could have a considerable acoustic significance [46]. We provide an interactive

Table 1. Bats used in the case studies of this paper.

Name	Scientific Name	# of Points
Tailless leaf-nosed bat	*Coelops frithii*	84,513
Greater spear-nosed bat	*Hipposideros commesoni*	126,646
Lyle's flying fox bat	*Pteropus lylei*	44,503
Intermediate horseshoe bat	*Rhinolophus affinis*	56,177
Woolly horseshoe bat	*Rhinolophus luctus*	91,524
Spectral vampire bat	*Vampyrum spectrum*	233,048

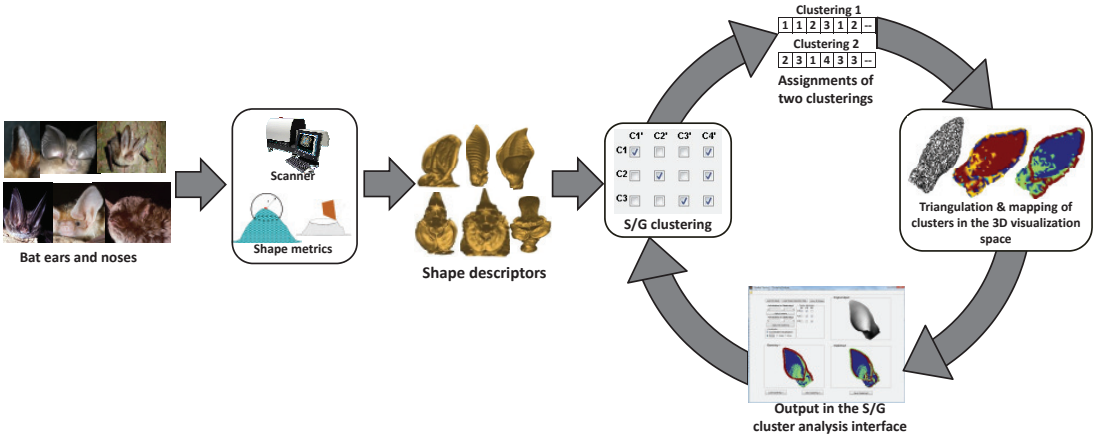


Figure 6. Analysis pipeline. Collected bat biosonar external organs are scanned using a 3D scanner, and shape descriptors are generated. The analyst then iteratively uses our interactive scatter/gather clustering interface to partition a biosonar system to study the shape. The user steers the clustering via a constraint table comprised of checkboxes.

tool to partition the biosonar systems into small parts based on local features and find correspondences of similar features in the biodiversity. This approach could lead experts to classify shapes as well as find correlation between these shapes and the function (e.g., beam pattern). Classification and correlation are beyond the scope of the current paper and we aim to provide those facilities with the developed tool in the future.

The ultimate goal is to identify common features of all available bat species in the database and characterize them based on their geometry and acoustic functions. Bat biosonar systems are not yet well studied and characterized in a way that maps geometry to acoustic functions. One of the objectives of this work is to partition bat biosonar systems based on the geometry and the local shape features that are responsible for the acoustic functions. The scatter/gather clustering approach described in this paper assists the expert in partitioning bat biosonar systems, understanding the local geometry, and deciding on which partitions represent common patterns (e.g., washboard, boundaries, ripples, ridges, and flat regions).

Fig. 6 shows the steps involved in our study of baffle shapes in the bat biosonar system. It shows that collected bat biosonar external organs are scanned using a 3D scanner, and shape descriptors are generated. Then the expert iteratively uses our interactive tool to partition a biosonar system to study the shape. Further details are provided in the following subsections.

4.2 Data Collection

There are more than 1,100 different species of bats known to science at present. In collaboration with local field biologists, specimens representing different bat species have been collected in location such as Cambodia, China, India, and Vietnam. The outer ears and noseleaves of these specimens were scanned using a high-resolution CT- scanner (Skyscan 1072 micro-CT) to create three-dimensional models repesented as 2D slices (images). The 2D slice images are then used to construct digital shape models in various formats for each outer ear and noseleaf. At present, the shape database compiled by the experts contains samples from about 105 bat species. For the case study in this paper, the experts used six different species shown in Table 1.

Local shape descriptors (LSD) [29] map a small *section* of the surface mesh around a vertex to an n-dimensional vector. This differs from *point* descriptors [14], such as curvature, which use just the surface data at the vertex. LSD tend to be more robust to noise in the mesh, and can also be scaled to capture features of different sizes. There are a variety of methods for calculating LSD; essentially, we sample the mesh in five concentric rings around the vertex, build up

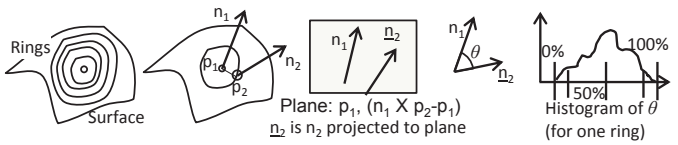


Figure 7. Generation of local shape descriptor.

a distribution of values for each ring for each vertex, then use PCA to reduce the dimension of this data. We use distributions in order to be orientation independent, and PCA to pull out the features of the distributions that are dominant.

More specifically, we sample five concentric rings around the vertex by intersecting spheres of increasing radii with the mesh, and sampling those rings uniformly. For each sample, we generate two numbers: the first is the angle change between the surface normal at the vertex and the surface normal at the sample point i, projected to the plane containing the two points and the vertex normal (θ_i). The second is the angle change between the current sample and the next one around the ring (ϑ_i). For each ring, we find the distribution in one of two ways. The first method is to sort the values (e.g., θ_i) and record the ones at the $0, 10, 50, 90, 100$ percentiles. The second method is to find the average and the standard deviation for the ring. Note that, for the latter, we use the Kubelka-Monk equation [39] to calculate distances between probability distribution functions and apply multidimensional scaling (MDS) on those distances instead of pure PCA. The methodology is shown in Fig. 7 in a sequence.

In all cases, each vertex of a dataset (bat ear or nose leaf) is described by a 20-dimensional vector generated by principal component analysis (PCA) or multidimensional scaling (MDS). Later we show that experts prefer to keep the number of clusters to less than five. For five clusters, the 20 components discovered by PCA or MDS are enough to capture the variability. We do not cluster the 3D points based on their orientation in the space; rather we partition each of the ears or noses using the 20-dimensional feature vector. That is, the datasets $\mathcal{X}$ and $\mathcal{X}'$ of Section 3.4 have $l_x = 20$ and are all based on the features generated by PCA or MDS — they do not contain 3D points. After our clustering framework is used, we map the clustering results to 3D points of the biosonar system with a color code for each cluster as an illustration of the results.

4.3 User Interface

The primary goals of the user interface are three-fold: (i) support iterative application of scatter-gather clustering so as to enable the user to fine-tune a clustering to their specific needs; and (ii) map clustering results involving 3D shape descriptors back onto the original 3D object so as to support direct manipulation; and (iii) support save and restore operations to enable the user to return to previous analysis as desired.

As illustrated in Fig. 8, the top left part of the user interface contains the *S/G Cluster Analysis Interface* where the user can select the number of clusters for k-means and S/G clustering using two sliders. The corresponding S/G constraint table is located in a small panel in a matrix with $k \times k'$ check boxes where k is the number of clusters in clustering 1 (k-means or given clustering resulting from a previous scatter-gather operation), and k' is the number of clusters in the desired clustering. The size of the matrix comprising the checkboxes changes with movements of the slider bars. Based on a discussion with the expert (see Section 5.1) on number of clusters, we allow a maximum

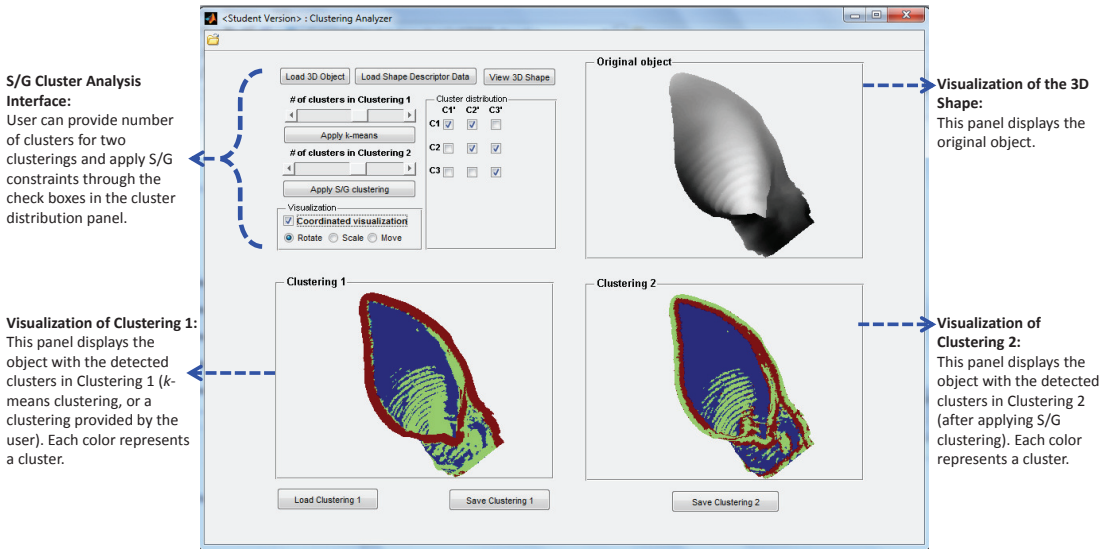


Figure 8. S/G cluster analysis user interface.

of five clusters in both clustering 1 and clustering 2 (by restricting the size of the two sliders in the top left part). The user checks the boxes as required to construct the S/G constraint table.

The other three panels shown in Fig. 8 are for the original 3D object, object with clustering 1 mapped in it, and object with clustering 2 mapped in it. The 3D object can contain hundreds of thousands of points (Table 1). Analysis of a large number of meshes in the object of the top right panel or partitioning them manually to understand the functionality is almost impossible for an expert. The interface we developed provides partitioning based on the locality of the shape descriptors of the points in the object. The panel at the bottom left shows the results of the current clustering (i.e., the initial k-means clustering or the results of a previous scatter-gather clustering).

Each color of the object surface indicates a cluster. Based on the clustering result in the bottom left panel, the user generally provides scatter/gather constraints in the top left part of the interface. The clustering outcome (clustering 2) with the scatter/gather constraints is shown in the bottom right panel. The user can then save the outcome shown in the bottom right panel, load it from the bottom left panel and start another iteration of the analysis.

In the top left part of the interface, the user also has an option to coordinate the visualizations between the original 3D object, clustering 1, and clustering 2. The user can rotate, scale, and move any of these three objects using the mouse. In the coordinated visualization, the user can interact with any one of the three visualizations but all three are affected. This allows the user to scrutinize the shapes and the clustering results. Each 3D object is created from the 3D points using Delaunay triangulation.

Although our user interface has focused on visualizations specific to the 3D geometry of the bat biosonar system, the underlying scatter-gather mathematical framework can be generalized to work from shape descriptors to other forms of data, so that visualizations suited to other applications can be substituted readily.

5 Results

In this section we describe the results of our scatter/gather clustering as analyzed by a specialist. The specialist is a researcher studying bat biosonar systems for the last three years. The specialist is also involved in the data collection process, digitization of the external biosonar organs, and analysis of the shapes.

In all subsections below, we use k to denote the number of clusters in the current clustering and k' to denote the clusters desired from the scatter/Gather clustering framework based on the constraints provided by the user. **The reader might get the illusion from some of the figures of this section that the cluster color-codes are merely changed in the second clustering. However, a closer look reveals key differences.** Moreover, our three-dimensional visualization tool allows the user to examine the results closely by allowing standard rotation, scaling, and translation facilities.

5.1 Configuration

Two main choices for investigation are selections of the ring radii and the number of clusters. In Section 4.2 we explained that we use different concentric rings around the vertex by intersecting a sphere of a certain radius with the mesh and sample that ring uniformly. The generated shape descriptor dataset varies with different ring radii. To examine which ring radii we should use to generate the shape descriptor data, we applied k-means clustering on the generated shape descriptors with radii 1%, 3%, 5%, 7%, and 9% (of the diameter of the entire mesh) for the ear of a tailless leaf-nosed bat. The expert preferred using the tailless leaf-nosed bat ear for this experiment because this ear is less complex than the ears of other bat species. The k-means clustering results for different radii are shown in Fig. 9, which shows that the boundaries are very narrow with low ring radii. The borders are thicker with larger radii, but cannot pick up fine details when the ring radius is too large. The expert preferred the results of the 5% ring radius because it was able to pick up two layers in the boundary as well as three sharp ridges (shown in red in the figure). After this preliminary investigation with the tailless leaf-nosed bat, the expert was provided with sample clustering results for the Lyle's flying fox bat and woolly horseshoe bat. The expert preferred 5% ring radii for these two species as well. For the rest of the results in this paper, we hence generated the shape descriptors with 5% ring radii.

Recall that our scatter/gather clustering algorithm is able to map k

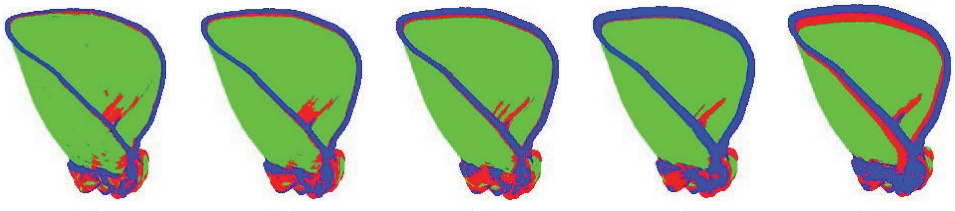
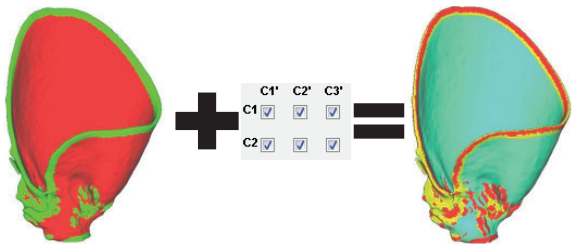
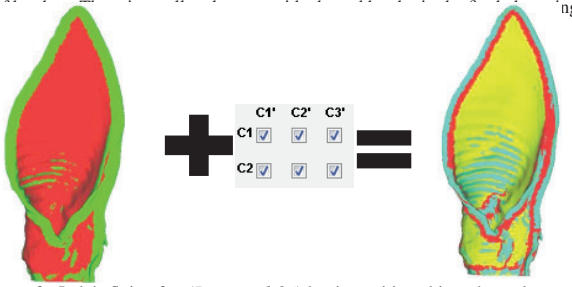


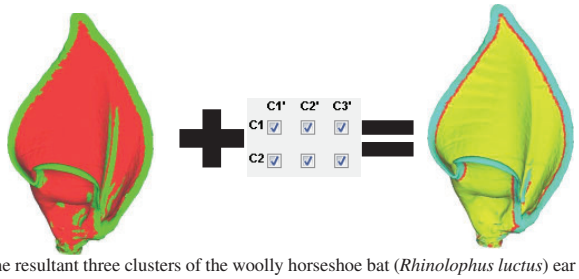
Figure 9. Comparison between clustering with different ring radii. (The ring radius is percentage of the bounding box diameter of the 3D mesh.)



(a) Two clusters of the ear of a tailless leaf-nosed bat (*Coelops frithii*) are repartitioned into three clusters using scatter/gather constraints. The resultant clustering provides two layers of [illegible]ng.



(b) An ear of a Lyle's flying fox (*Pteropus lylei*) bat is partitioned into three clusters from two groups. Two layers of the pinna boundary are revealed as well as a better washboard pattern. The washboard pattern and the outer pinna boundary of the final clustering fall into the same cluster (cluster in green).



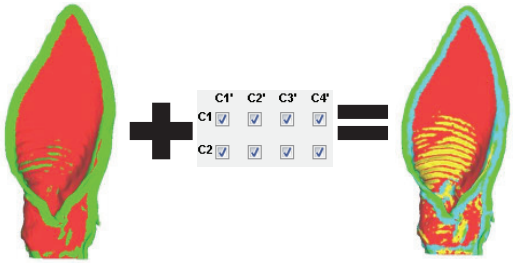
(c) The resultant three clusters of the woolly horseshoe bat (*Rhinolophus luctus*) ear reveal two layers in the boundary.

Figure 10. Better boundaries using scatter/gather constraints to form three clusters from two. In all the cases shown here, the boundaries are better partitioned after scatter/gather clustering.

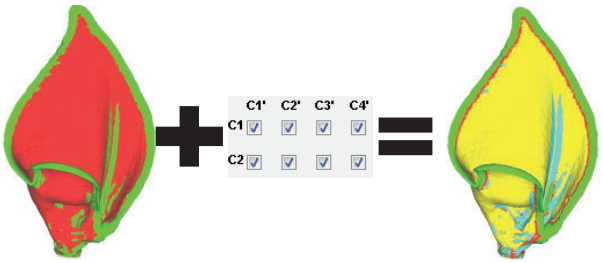
clusters into k' clusters, where k' can be smaller than, equal to, or larger than k. We observed that the specialist was mostly interested in two to four clusters. When he was specifically asked to provide an explanation of the number of clusters, he mentioned that he was interested in several regions of bat ears and noses, e.g., borders, washboard patterns, ripples, ridges, and flat regions. Not all of the bats have all these regions in their biosonar systems. Based on this input from the specialist, we provided a choice of two to five clusters for both k-means and scatter/gather clustering. That is, the scatter/gather constraint table can be a $k \times k'$ matrix where each k and k' can take any integer value from two to five.

5.2 Scatter/Gather Clustering with $k < k'$

Fig. 10 shows a few examples of scatter/gather clustering with $k < k'$. The figure illustrates partitionings for ears of three different bat species: (a) tailless leaf-nosed bat, (b) Lyle's flying fox bat, and (c) woolly horseshoe bat. The number of clusters in the given clustering is $k = 2$, and the number of clusters in the output clustering is $k' = 3$. For each bat ear, the user provided scatter/gather constraints in a 2×3 matrix with all cells checked indicating that the user expects construction of three clusters from two where each of the three clusters of the final clustering can contain points from any of the two given k-means clusters. For all three bat species, our scatter/gather clustering framework picked up two layers of borders unlike the corresponding k-means clustering. The expert provided a detailed explanation of the partitionings obtained for Lyle's flying fox bat ear. In the case of Lyle's flying fox bat (Fig. 10(b)), the scatter/gather clustering provided better washboard patterns than the k-means clustering. Additionally, the washboard patterns found with scatter/gather clustering are in the same



(a) An ear of a Lyle's flying fox (*Pteropus lylei*) bat is partitioned into four clusters from two groups. Two layers of the pinna boundary are revealed as well as a better washboard pattern. The washboard patterns are in a separate cluster (yellow) from the outer pinna boundary unlike Fig. 10(b).



(b) Two clusters of the ear of a woolly horseshoe bat (*Rhinolophus luctus*) are partitioned into four clusters using scatter/gather constraints. The resultant clustering provides two layers of borders (green and red), a separated vertical ridge (light blue), and the rest of the ear (yellow).

Figure 11. Better partitioning with scatter/gather constraints from two clusters to four. In both the cases shown here, the boundaries are better partitioned and some regions are well separated after the scatter/gather clustering is applied.

cluster as the outer pinna boundary (cluster with green color). The features that were not prominent in the k-means clustering result became apparent in the partitioning discovered by our scatter/gather clustering approach.

Fig. 11 shows two examples with a Lyle's flying fox bat ear and a woolly horseshoe bat ear. In each case, $k = 2$ and $k' = 4$. These two results were generated when the expert was analyzing the shapes to obtain details from two clusters provided by k-means. The expert provides a uniform scatter/gather constraint table for both cases. In the case of Lyle's flying fox bat (Fig. 11(a)), the washboard patterns were separated in one cluster (yellow). Note that in the scatter/gather clustering of Fig. 10(b) the washboard patterns were clustered together with the outer pinna boundary, but the washboard patterns are in a separate cluster in the scatter/gather clustering shown in Fig. 11(a). This indicates that scatter/gather clustering is able to provide finer details with larger k'.

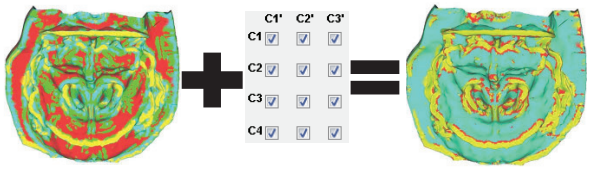
Fig. 11(b) shows that in addition to the two layers (red and green) in the border of the ear of the woolly horseshoe bat, there is a separate cluster for the vertical ridge (light blue) and a cluster for the rest of the ear (yellow). The boundaries and the vertical ridges were in the same cluster (green) in the k-means clustering but they are well separated (green, red, and light blue) in the scatter/gather clustering result. This helps the expert in characterizing the regions of the bat ears based on the local shapes.

The expert provided us with comments regarding the boundary regions:

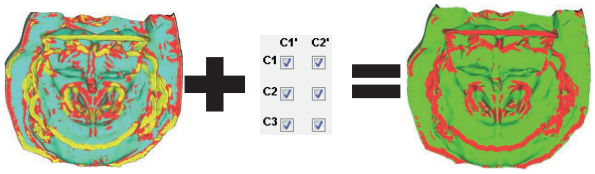
> " *The 'pattern border' of any shape if it exists is better visible in the scatter/gather clustering results. By 'pattern border' I mean if the border is having two different colors — Mostly one color is sandwiched in between two or more colors. This could be useful for us if we want to isolate the border of the ear and reduce the intensity of local features like washboard pattern and ridges by smoothing them out and studying the resultant beam patterns.*"

In addition to this, his comment regarding scatter/gather clustering with $k < k'$ is:

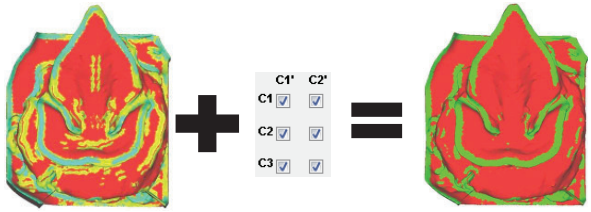
> "*The resulting clusters are able to isolate regions with similar local features like washboard pattern and ridge in separate clusters, especially for ears.*"



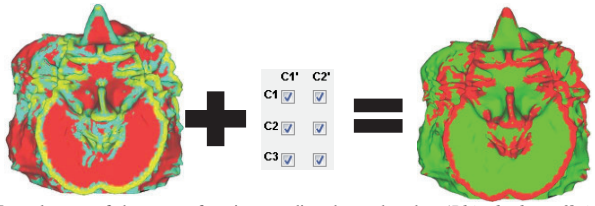
(a) Four clusters of the nose of a giant leaf-nosed bat (*Hipposideros commersoni*) are gathered together into three clusters. The borders are better separated in the resultant clustering.



(b) Three clusters of the nose of a giant leaf-nosed bat are combined together to form two clusters. The borders are obtained in one cluster in the resulting clustering.



(c) Three clusters of the nose of a greater spear-nosed bat (*Hipposideros commesoni*) are constrained to form two clusters. The resulting two clusters clearly separate the border from the rest of the nose.



(d) Two clusters of the nose of an intermediate horseshoe bat (*Rhinolophus affinis*) are obtained from three clusters. The obtained clustering (with two clusters) provides better separation of the borders.

Figure 12. Scatter/gather clustering (with $k > k'$) applied on several species to combine overpartitioned clusters.

5.3 Scatter/Gather Clustering with $k > k'$

In our study, we observed that the analyst sometimes desires to merge clusters because the existing clustering overpartitioned the data. A sample is shown in Fig. 12. Fig. 12(a) shows that a giant leaf-nosed bat nose has been partitioned into four clusters and the user provides scatter/gather constraints to obtain three clusters. The borders are better separated (yellow) in the obtained clustering. During the analysis, the user also used the scatter/gather constraints to obtain two clusters from three (Fig. 12(b)). The two obtained clusters of Fig. 12(b)(right) reveal the border of the nose more clearly than the three clusters shown at left. Fig. 12(c) and (d) both show that the borders are better discovered when scatter/gather constraints are applied to obtain two clusters from three clusters.

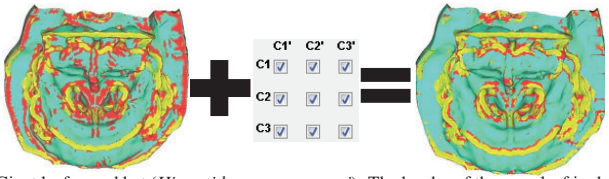
Each of the samples shown in Fig. 12 initially had complex partitions with a larger number of clusters and the user attempted to merge them to obtain a simpler partitioning. Scatter/gather clustering provides an abstraction of many clusters with $k > k'$.

The expert's comment on scatter/gather clustering with $k > k'$ is:

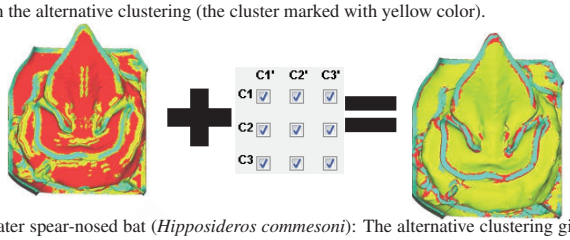
"*The resulting clusters have better noise isolation in terms of identifying the borders, especially in nose leaves.*"

5.4 Scatter/Gather Clustering with $k = k'$

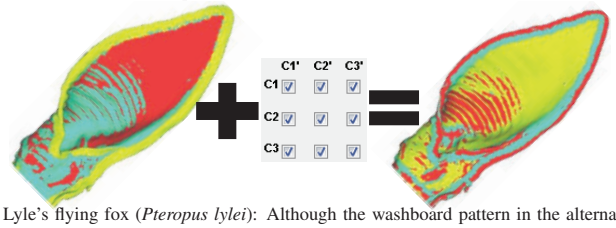
In sections 5.2 and 5.3, we described how the scatter/gather clustering framework can help in analyzing partitions by providing a facility to split and/or merge clusters. In this section, we show a special case of scatter/gather clustering where the number of clusters remains the same in the obtained clustering but the outcome is as disparate as possible from the given (k-means) clustering, which we described



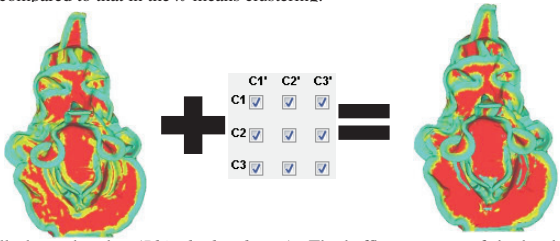
(a) Giant leaf-nosed bat (*Hipposideros commersoni*): The border of the nose leaf is shown better in the alternative clustering (the cluster marked with yellow color).



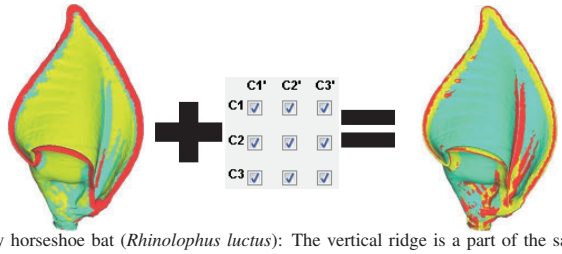
(b) Greater spear-nosed bat (*Hipposideros commesoni*): The alternative clustering gives a better view of the baffle shape.



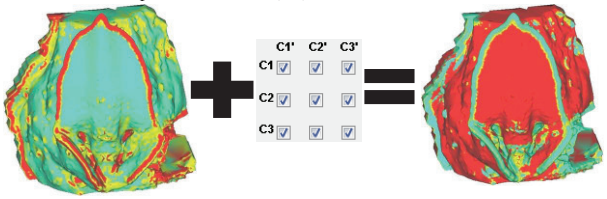
(c) Lyle's flying fox (*Pteropus lylei*): Although the washboard pattern in the alternative clustering has the same color as the border of the ear, the washboard pattern is better isolated compared to that in the k-means clustering.



(d) Woolly horseshoe bat (*Rhinolophus luctus*): The baffle geometry of the border and relatively flat regions of the nose leaf are more clearly distinguished in the alternative clustering.



(e) Woolly horseshoe bat (*Rhinolophus luctus*): The vertical ridge is a part of the same cluster as the border of the ear. The ridge is one of the important structures of the ear that defines the over all shape of the receiver (ear).



(f) Spectral vampire bat (*Vampyrum spectrum*): The baffle geometry of the border of the nose leaf are more clearly distinguished in the alternative clusterings.

Figure 13. Some illustrative results with the special case of scatter/gather clustering (known as alternative clustering). The comments from an expert are inline.

as "alternative clustering" in Section 3.5. Alternative clustering is sometimes important to an analyst to view partitionings from multiple perspectives. Alternative clustering is known to capture less prominent features that could be missed by conventional k-means clustering. Since bats have evolved for more than a million years, their biosonar systems have regions with different levels of prominence. We observed that the experts sometimes provide uniform S/G constraint

'square' tables to find hidden or less prominent layers.

Fig. 13 shows scatter/gather clustering results with $k = k'$ for ear or nose of five different species: (a) giant leaf-nosed bats, (b) greater spear-nosed bat, (c) Lyle's flying fox bat, (d and e) woolly horseshoe bat, and (f) spectral vampire bat.

Fig. 13(a) shows that the borders of the nose leaf of the giant leaf-nosed bats are better isolated in the clustering obtained by scatter/gather framework. Also in case of the nose leaf of the greater spear-nosed bat (Fig. 13(b)), the baffle shape is better in the scatter/gather clustering. The scatter/gather clustering of the ear of the Lyle's flying fox bat (Fig. 13(c)) provides two layers in the boundary and clear washboard patterns. The baffle geometry of the border and relatively flat regions of the nose leaf of the woolly horseshoe bat are more clearly distinguished in the scatter/gather clustering compared to the given k-means clusters (Fig. 13(d)). The scatter/gather clustering of Fig. 13(e) picks up the important vertical ridge of the woolly horseshoe bat pinna. The border of the spectral vampire bat nose leaf is clearly distinguished and the noise level is reduced in the scatter/gather clustering shown in Fig. 13(f). Overall, scatter/gather clustering with $k = k'$ provides an alternative partitioning of a given one. In all the cases in addition to finding an alternative partitioning, the expert reported that the noise level was reduced in the alternative clustering.

The expert's comment on our scatter/gather clustering with $k = k'$ is:

" *I find that alternative clusters are good at isolating and reducing noise compared to k-means. It also reveals interesting regions that are less prominent in k-means.* "

5.5 Iterative Scatter/Gather Clustering using Sparse Contingency Tables

Here we present an interactive scenario where the expert uses scatter/gather clustering to obtain a desired partitioning by refining it several times. The expert is trying to find partitions of a woolly horseshoe bat ear. The expert at first partitions the object into two clusters using k-means clustering (Fig. 1(a)). The expert finds the partitions interesting. He observes that the boundary and the vertical ridges are in the same cluster (green), and the rest of the ear is in another cluster. This fosters a thought in the expert's mind that the vertical ridges could be separated to form a new cluster. The expert also believes that there could be less prominent layers in the borders of the ear. Being unsure about the constraints, the expert provides a uniform scatter/gather constraint table of size 2×3 indicating that he desires three clusters out of the two clusters. Our scatter/gather clustering provides the result shown in Fig. 1(b). The partitioning of Fig. 1(b) was able to pick up two border layers, but the vertical ridges now diminish inside the surrounding cluster. At this point, the expert believes that it is more important to reveal the shape of the vertical ridges rather than discovering the layers in the boundary. The expert now provides an S/G constraint table to merge two boundaries (light blue and red), and split the mid region of the ear (yellow) into two clusters. The resulting clusters are shown in Fig. 1(c) where the vertical ridges are well separated in one cluster. The expert now desires to split the border into two layers that he previously merged. Setting up an S/G constraint table of size 3×4 as shown in the middle of (c) and (d) objects of Fig. 1, the user obtains four clusters. These four clusters contain two layers of border (green and red), vertical ridges (light blue), and the flat region of the ear (yellow).

6 Other Interaction Styles and Extensions

We now outline some limitations to our current implementation and possibilities for future work. The binary checboxes used in our scatter-gather constraint table enforce an all-or-none distribution of points between clusters, i.e., the user cannot enforce a certain percentage of data points from a cluster to be scattered or gathered. Mathematically, this capability is easy to support since all our framework requires is a normalized contingency table. Recall that we normalize the constraint table so as to distribute the total probability mass uniformly across all the columns/rows checked. If the user has specific feedback, that information can be used to reweight the contingency table. All that is required is that the row-sums and column-sums (marginals) be normalized to sum upto 1. Design of user interfaces that can elicit such detailed feedback from the user is a direction of future work.

Secondly, with a large number of clusters, the scatter/gather constraint table can grow unwieldly. For the application described here, the number of clusters is restricted to a small number, because experts were primarily interested in discovering features corresponding to a few regions of bat ears/noses, e.g., borders, washboard patterns, ripples, ridges, and flat regions. For other applications (e.g., document clustering, gene clustering, image segmentation), constraint tables can become larger and difficult to fill out. One option we are exploring is to provide higher level support for filling in the constraint table, e.g., filling out the table diagonally, filling the checkboxes row or column wise, and arbitrarily focusing on sub-boxes to fill out a subset of neighboring cells. We are also exploring avenues for gathering implicit feedback about cluster restructurings from the user.

Finally, in clustering applications, user constraints can be provided either at the instance-level or at the cluster level. With the instance-level constraints, the user can directly manipulate the assignment of the data points into clusters but with high-dimensional datasets, as the number of data points in a cluster becomes large, direct instance-level constraints might become cumbersome. The scatter/gather constraints described in this paper are cluster-level constraints (i.e., clusters being broken up, clusters being aggregated) and can be considered as an abstraction of the instance-level feedback. Determining how to effectively incorporate user feedback at two different levels of abstraction is a direction of future work.

7 Conclusion

We have described a novel approach to steer clustering results and demonstrated its application to studying baffle shapes in the bat biosonar system. Although we have not focused on this aspect here, it is possible to plug-and-play many different clustering algorithms inside the scatter/gather framework. In [31], we have shown how spectral clustering, coclustering, and clustering with instance-level constraints can be used along with a contingency table framework. The only requirement is that the clusters be defined via proximity to prototypes (which subsumes a large class of vector quantization algorithms); the prototypes are then the variables that are optimized w.r.t. the scatter/gather contingency table.

Our studies with users have revealed that domain experts are adept at supplying scatter/gather tables and able to iteratively use them to obtain desired outcomes. We also aim to explore additional applications of our framework to new domains, and to identify more expressive forms of user feedback that can be incorporated into our approach. We also aim to characterize each cluster in the partitions and employ automatic enrichment algorithms to classify new bat ears and noses.

Acknowledgments

This work is supported in part by the Institute for Critical Technology and Applied Science — Virginia Tech, the US National Science Foundation through grants CCF-0937133, CCF-0702662, DBI-451069, and DBI-1053171, the US Army Research Office (award id 451069), CRA Distributed Mentor Program, AFRL through grant FA8650-09-2-3938, and AFOSR through grant FA9550-09-1-0153.

References

[1] S. R. Aghabozorgi and T. Y. Wah. Recommender Systems: Incremental Clustering on Web Log Data. In *ICIS '09*, pages 812–818, 2009.

[2] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. *SIGMOD Rec.*, 27(2):94–105, 1998.

[3] Z. Ahmed, P. Yost, A. McGovern, and C. Weaver. Steerable Clustering for Visual Analysis of Ecosystems. In *EuroVA '11*, pages 49–52, 2011.

[4] O. Alonso and J. Talbot. Structuring Collections with Scatter/Gather Extensions. In *SIGIR '08*, pages 697–698, 2008.

[5] G. Andrienko and N. Andrienko. Interactive Cluster Analysis of Diverse Types of Spatiotemporal Data. *SIGKDD Explor. Newsl.*, 11(2):19–28, 2010.

[6] G. Andrienko, N. Andrienko, S. Rinzivillo, M. Nanni, D. Pedreschi, and F. Giannotti. Interactive Visual Clustering of Large Collections of Trajectories. In *VAST '09*, pages 3–10, 2009.

[7] M. Ankerst, S. Berchtold, and D. A. Keim. Similarity Clustering of Dimensions for an Enhanced Visualization of Multidimensional Data. In *INFOVIS '98*, pages 52–60, 1998.

[8] J. Bernard, T. von Landesberger, S. Bremm, and T. Schreck. Cluster Correspondence Views for Enhanced Analysis of SOM Displays. In *VAST '10*, pages 217–218, 2010.

[9] S. Brohee and J. van Helden. Evaluation of Clustering Algorithms for Protein-protein Interaction Networks. *BMC Bioinformatics*, 7:488, 2006.

[10] R. Caruana, M. Elhawary, N. Nguyen, and C. Smith. Meta Clustering. In *ICDM '06*, pages 107–118, 2006.

[11] K. Chen and L. Liu. iVIBRATE: Interactive Visualization-based Framework for Clustering Large Datasets. *ACM Trans. Inf. Syst.*, 24(2):245–294, 2006.

[12] C. Cheng, A. W. Fu, and Y. Zhang. Entropy-based Subspace Clustering for Mining Numerical Data. In *KDD '99*, pages 84–93, 1999.

[13] J. Choo, S. Bohn, and H. Park. Two-stage Framework for Visualization of Clustered High Dimensional Data. In *VAST '09*, pages 67–74, 2009.

[14] C. S. Chua and R. Jarvis. Point Signatures: A New Representation for 3D Object Recognition. *Int. J. Comput. Vision*, 25(1):63–85, 1997.

[15] T. Coleman and Y. Li. An Interior, Trust Region Approach for Nonlinear Minimization Subject to Bounds. *SIAM Journal on Optimization*, 6:418–445, 1996.

[16] Y. Cui, X. Fern, and J. G. Dy. Non-redundant Multi-view Clustering via Orthogonalization. In *ICDM '07*, pages 133–142, 2007.

[17] D. R. Cutting, D. R. Karger, and J. O. Pedersen. Constant Interaction-time Scatter/Gather Browsing of Very Large Document Collections. In *SIGIR '93*, pages 126–134, 1993.

[18] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/Gather: a Cluster-based Approach to Browsing Large Document Collections. In *SIGIR '92*, pages 318–329, 1992.

[19] X. Dang and J. Bailey. A Hierarchical Information Theoretic Technique for the Discovery of Non-linear Alternative Clusterings. In *KDD '10*, pages 573–582, 2010.

[20] X. Dang and J. Bailey. Generation of Alternative Clusterings Using the CAMI Approach. In *SDM '10*, pages 118–129, 2010.

[21] I. Davidson and Z. Qi. Finding Alternative Clusterings Using Constraints. In *ICDM '08*, pages 773–778, 2008.

[22] I. Davidson and S. S. Ravi. Clustering with Constraints: Feasibility Issues and the k-Means Algorithm. In *SDM '05*, pages 201–211, 2005.

[23] I. Davidson, S. S. Ravi, and M. Ester. Efficient Incremental Constrained Clustering. In *KDD '07*, pages 240–249, 2007.

[24] M. desJardins, J. MacGlashan, and J. Ferraioli. Interactive Visual Clustering. In *IUI '07*, pages 361–364, 2007.

[25] D. Gondek and T. Hofmann. Non-redundant Clustering with Conditional Ensembles. In *KDD '05*, pages 70–77, 2005.

[26] G. Govaert and M. Nadif. Clustering with Block Mixture Models. *PR*, 36(2):463–473, 2003.

[27] M. Greenacre. Clustering the Rows and Columns of a Contingency Table. *J. of Classification*, 5(1):39–51, 1988.

[28] D. Guo, J. Chen, A. MacEachren, and K. Liao. A Visualization System for Space-Time and Multivariate Patterns (VIS-STAMP). *TVCG*, 12(6):1461 –1474, 2006.

[29] P. Heider, A. Pierre-Pierre, R. Li, and C. Grimm. Local Shape Descriptors, A Survey and Evaluation. In *Eurographics Workshop on 3D Object Retrieval*, pages 49–57, 2011.

[30] M. Höferlin, B. Höferlin, D. Weiskopf, and G. Heidemann. Interactive Schematic Summaries for Exploration of Surveillance Video. In *ICMR '11*, pages 9:1–9:8, 2011.

[31] M. S. Hossain. *Exploratory Data Analysis using Clusters and Stories*. PhD thesis, Virginia Tech, Blacksburg, VA, June 2012.

[32] M. S. Hossain, S. Tadepalli, L. T. Watson, I. Davidson, R. F. Helm, and N. Ramakrishnan. Unifying Dependent Clustering and Disparate Clustering for Non-homogeneous Data. In *KDD '10*, pages 593–602, 2010.

[33] Y. Huang and T. M. Mitchell. Text Clustering with Extended User Feedback. In *SIGIR '06*, pages 413–420, 2006.

[34] I. Hwang, M. Kahng, and S.-g. Lee. Exploiting User Feedback to Improve Quality of Search Results Clustering. In *ICUIMC '11*, pages 68:1–68:5, 2011.

[35] D. H. Jeong, A. Darvish, K. Najarian, J. Yang, and W. Ribarsky. Interactive Visual Analysis of Time-series Microarray Data. *Vis. Comput.*, 24(12):1053–1066, 2008.

[36] G. Jones and E. C. Teeling. The Evolution of Echolocation in Bats. *Trends in Ecology & Evolution*, 21(3):149 – 156, 2006.

[37] Y. Kanada. Axis-specified Search: a Fine-grained Full-text Search Method for Gathering and Structuring Excerpts. In *DL '98*, pages 108–117, 1998.

[38] S. Kaski, J. Nikkilä, J. Sinkkonen, L. Lahti, J. E. A. Knuuttila, and C. Roos. Associative Clustering for Exploring Dependencies between Functional Genomics Data Sets. *IEEE/ACM TCBB*, 2(3):203–216, 2005.

[39] P. Kubelka. New Contributions to the Optics of Intensely Light-Scattering Materials. Part I. *J. Opt. Soc. Am.*, 38(5):448–448, 1948.

[40] T. Li, C. Ding, and M. I. Jordan. Solving Consensus and Semi-supervised Clustering Problems Using Nonnegative Matrix Factorization. In *ICDM '07*, pages 577–582, 2007.

[41] J. Liang, B. Abidi, and M. Abidi. Automatic X-ray Image Segmentation for Threat Detection. In *ICCIMA '03*, pages 396 – 401, 2003.

[42] J. Ma and R. Müller. A Method for Characterizing the Biodiversity in Bat Pinnae as a Basis for Engineering Analysis. *Bioinspiration & Biomimetics*, 6(2):026008, 2011.

[43] G. Miao, J. Tatemura, W.-P. Hsiung, A. Sawires, and L. E. Moser. Extracting Data Records from the Web using Tag Path Clustering. In *WWW '09*, pages 981–990, 2009.

[44] R. Müller. Numerical Analysis of Biosonar Beamforming Mechanisms and Strategies in Bats. *J Acoust Soc Am.*, 128(3):1414–1425, 2010.

[45] S. Monti, P. Tamayo, J. Mesirov, and T. Golub. Consensus Clustering: A Resampling-Based Method for Class Discovery and Visualization of Gene Expression Microarray Data. *Machine Learning*, 52:91–118, 2003.

[46] R. Müller. A Numerical Study of the Role of the Tragus in the Big Brown Bat. *J Acoust Soc Am*, 116(6):3701–12, 2004.

[47] M. Nadif and G. Govaert. Block Clustering of Contingency Table and Mixture Model. In *IDA '05*, pages 249–259, 2005.

[48] E. J. Nam, Y. Han, K. Mueller, A. Zelenyuk, and D. Imre. ClusterSculptor: A Visual Analytics Tool for High-Dimensional Data. In *VAST '07*, pages 75–82, 2007.

[49] D. Niu, J. G. Dy, and M. I. Jordan. Multiple Non-redundant Spectral Clustering Views. In *ICML '10*, pages 831–838, 2010.

[50] M. K. Obrist, M. B. Fenton, J. L. Eger, and P. A. Schlegel. What Ears do for Bats: a Comparative Study of Pinna Sound Pressure Transformation in Chiroptera. *J Exp Biol*, 180:119–152, 1993.

[51] V. A. Petrushin. Mining Rare and Frequent Events in Multi-camera Surveillance Video using Self-organizing Maps. In *KDD '05*, pages 794–800, 2005.

[52] Z. Qi and I. Davidson. A Principled and Flexible Framework for Finding Alternative Clusterings. In *KDD '09*, pages 717–726, 2009.

[53] D. Russo, G. Jones, and R. Arlettaz. Echolocation and Passive Listening by Foraging Mouse-eared Bats Myotis myotis and M. blythii. *J Exp Biol*, 210(1):166–176, 2007.

[54] H. Schnitzler and E. Kalko. *Bat Biology and Conservation*. Washington, DC: Smithsonian Institution Press, 1998.

[55] T. Schreck, J. Bernard, T. Tekusova, and J. Kohlhammer. Visual Cluster Analysis of Trajectory Data with Interactive Kohonen Maps. In *VAST '08*, pages 3 –10, 2008.

[56] J. Sese, Y. Kurokawa, M. Monden, K. Kato, and S. Morishita. Constrained Clusters of Gene Expression Profiles with Pathological Features. *Bioinformatics*, 20(17):3137 – 3145, 2004.

[57] J. Sinkkonen, S. Kaski, and J. Nikkilä. Discriminative Clustering: Optimal Contingency Tables by Learning Metrics. In *ECML '02*, pages 418–430, 2002.

[58] J. Sinkkonen, J. Nikkilä, L. Lahti, and S. Kaski. Associative Clustering. In *ECML '04*, pages 396–406, 2004.

[59] A. Strehl and J. Ghosh. Cluster Ensembles — a Knowledge Reuse Framework for Combining Multiple Partitions. *JMLR*, 3:583–617, 2003.

[60] K. Wagstaff and C. Cardie. Clustering with Instance-level Constraints. In *ICML '00*, pages 1103–1110, 2000.

[61] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained K-means Clustering with Background Knowledge. In *ICML '01*, pages 577–584, 2001.

[62] X. Wang and I. Davidson. Flexible Constrained Spectral Clustering. In *KDD '10*, pages 563–572, 2010.

[63] Y. Xu and V. O. an Dong Xu. Clustering Gene Expression Data using a Graph-theoretic Approach: an Application of Minimum Spanning Trees. *Bioinformatics*, 18(4):536–545, 2002.

[64] Y. Zeng, J. Tang, J. Garcia-Frias, and G. R. Gao. An Adaptive Meta-Clustering Approach: Combining the Information from Different Clustering Results. In *CSB '02*, pages 276–287, 2002.