Towards a Systematic Combination of Dimension Reduction and Clustering in Visual Analytics

John Wenskovitch, *Student Member, IEEE*, Ian Crandell, Naren Ramakrishnan, *Member, IEEE*, Leanna House, Scotland Leman, Chris North

Abstract— Dimension reduction algorithms and clustering algorithms are both frequently used techniques in visual analytics. Both families of algorithms assist analysts in performing related tasks regarding the similarity of observations and finding groups in datasets. Though initially used independently, recent works have incorporated algorithms from each family into the same visualization systems. However, these algorithmic combinations are often ad hoc or disconnected, working independently and in parallel rather than integrating some degree of interdependence. A number of design decisions must be addressed when employing dimension reduction and clustering algorithms concurrently in a visualization system, including the selection of each algorithm, the order in which they are processed, and how to present and interact with the resulting projection. This paper contributes an overview of combining dimension reduction and clustering into a visualization system, discussing the challenges inherent in developing a visualization system that makes use of both families of algorithms.

Index Terms—Dimension reduction, clustering, algorithms, visual analytics.

1 Introduction

Visual metaphors for exploring high-dimensional datasets come in a variety of forms, each with their own strengths and weaknesses in both visualization and interaction [37, 69]. In particular, datasets with high dimensionality present tractability challenges for computation, design, and interaction [29]. One frequently used method of visual abstraction is to reduce a high-dimensional dataset into a low-dimensional space while preserving properties of the high-dimensional structure (e.g., retain or respect pairwise relationships from the higher dimensions in the lower dimensional projection). Such dimension reduction algorithms are useful abstractions because some of the dimensions in the dataset may not be essential to understanding the underlying patterns in the dataset [38]. Instead, a subset of the dimensions can be selected or learned (or new dimensions introduced) to define the important characteristics of the dataset. The visualization tasks associated with dimension reduction algorithms have been well studied [14, 15].

Many dimension reduction algorithms employ a "proximity $\approx$ similarity" metaphor, in which a distance function measures the similarity of pairs of observations[1] at the high-dimensional level and attempts to preserve those distance relationships in the low-dimensional projection by minimizing a stress function. Due to this "proximity $\approx$ similarity" relationship, observations with high similarity or an underlying relationship can form implicit clusters in the low-dimensional projection. Indeed, clustering can even be thought of as extremely low-resolution dimension reduction, where knowledge about the various attributes of the observations leads to a one-dimensional bin assignment (or a set of probabilities for bin assignments). This relationship between dimension reduction and clustering is also supported mathematically in specific instances. For example, Ding and He [27] proved that principal components are the continuous solutions to the discrete cluster membership indicators for k-means clustering, indicating that Principal Component Analysis (PCA) dimension reduction implicitly performs data clustering as well.

Indications from previous studies [8, 33] have shown that analysts use a complex combination of both developing clusters and organizing observations in space in the sensemaking process [76] as they explore a dataset. These explorations generate clusters created by the analyst during *exploratory* interactions to spatially organize information on the display, as well as clusters that naturally develop due to *expressive* interactions updating the underlying layout (these interaction types are defined by Endert et al [34]). Other studies have also linked dimension reduction algorithms to clustered data; for example, Choo et al. discusses dimension reduction methods for two-dimensional visualization of high-dimensional clustered data, proposing a two-stage framework for visualizing such data based on dimension reduction methods [21].

While dimension reduction algorithms and clustering algorithms have been implemented together in a number of visualization systems, these algorithms often operate independently and in parallel. In other words, each algorithm supports some analysis component in the system without the influence of the other algorithm: perhaps a collection of observations are clustered, but the output of that clustering has limited or no effect on the dimension-reduced layout of the observations. Alternatively, a change to the spatialization may perceptually imply the need for a change to the cluster assignment, but no update to the cluster assignment may occur. The second case can be seen in the iVisClustering system [57]. This tool clusters documents into a collection of topics and uses a force-directed layout in the Cluster Relation View to present the documents spatially. However, making a change to the layout of the projection (see Fig. 1) has no effect on the clustering assignments of the documents.

Exploring the connections between dimension reduction and clus-

- *John Wenskovitch, Naren Ramakrishnan, and Chris North are with the Virginia Tech Department of Computer Science. E-mails: {jw87 | naren | north}@cs.vt.edu.*
- *Ian Crandell, Leanna House, and Scotland Leman are with the Virginia Tech Department of Statistics. E-mails: {ian85 | lhouse | leman}@vt.edu.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

[1] In this work, we employ the convention of referring to the features of a dataset as *dimensions*, individual data items as *observations*, and the features of those observations as *attributes*.

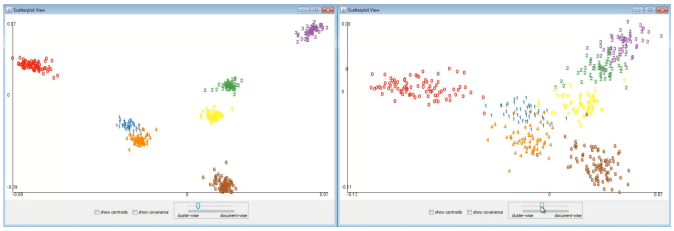


Fig. 1. The iVisClustering system [57] incorporates dimension reduction and clustering algorithms in the same system; however, making a change to the layout has no effect on the clustering assignment.

Table 1. A selection of dimension reduction algorithms, organized by the complexity of manifold each can learn: linear manifolds, nonlinear manifolds, and algorithms that have implementations of both types.

	Selected Dimension Reduction Algorithms
Linear	Factor Analysis [43]
	Principal Component Analysis (PCA) [74]
	Probabilistic PCA (PPCA) [84]
	Projection Pursuit [40]
Both	Feature Selection [42]
	Independent Component Analysis (ICA) [49]
	Multidimensional Scaling (MDS) [85]
	Weighted MDS (WMDS) [18]
Nonlinear	Glimmer [50]
	Isomap [82]
	Latent Dirichlet Allocation (LDA) [11]
	t-Distributed Stochastic Neighbor Embedding (t-SNE) [65]

tering algorithms leads to several natural research questions. If the data separates into implicit clusters, and the analyst sees advantages in the creation of these implicit clusters, can we appropriately support explicit cluster definitions so that the dimension reduction and clustering algorithms support each other rather than conflict with each other (or simply do not interact with each other)? If so, how should we define, visualize, and interact with both observations and clusters in a dimension-reduced projection? And finally, is there a difference between how analysts interpret and interact with low-dimensional clusters as opposed to high-dimensional clusters?

Our research explores initial steps to address these questions. In particular, this work includes the following contributions:

1. An overview of combining dimension reduction and clustering techniques into a visualization system, including a discussion of algorithms, tasks, visualizations, and interactions.

2. A discussion of the design decisions that must be addressed when creating a visualization system that combines dimension reduction and clustering algorithms.

The remainder of this paper discusses these contributions through the exploratory data analysis process. We begin by providing an overview of existing dimension reduction and clustering algorithms in Sect. 2. From there, we discuss common high-dimensional data analysis tasks is Sect. 3, visualizations to support those tasks in Sect. 4, and interactions on those visualizations in Sect. 5. We close with a discussion of further challenges and lessons learned in Sect. 6 and conclude in Sect. 7 with a summary of design questions that should be considered when developing a tool combining these algorithm families.

2 Algorithms

In this section, we summarize the variety of algorithms that address dimension reduction and clustering tasks in visualization systems.

2.1 Dimension Reduction Algorithms

The goal of dimension reduction algorithms is to represent high-dimensional data in a low-dimensional space while preserving high-dimensional structures, including outliers and clusters [58]. Dimension reduction has a scalability advantage over other methods for visualizing high-dimension data such as parallel coordinate plots and heatmaps, but with the disadvantage of information loss when transforming the data into the low-dimensional projection [37, 61, 69]. Here, we summarize many of the common dimension reduction algorithms in the Visualization field; more detailed surveys of dimension reduction algorithms can be found in the literature [38, 39, 58, 91]. In addition, several tools have been implemented that allow analysts to switch between and compare dimension reduction algorithms [62, 77].

Dimension reduction algorithms can be divided into linear and nonlinear classes, referring to the structure of the underlying manifolds or topological spaces that each class can learn. Linear dimension reduction algorithms are limited to learning linear manifolds, while nonlinear dimension reduction algorithms can learn more complex manifolds. Still other dimension reduction algorithms have been implemented in both linear and nonlinear variants. Table 1 provides a set of commonly used dimension reduction algorithms in the visualization literature, divided into whether they are linear, nonlinear, or have implementations of both levels of manifold complexity. Principal Component Analysis (PCA) is perhaps the most frequently-used linear dimension reduction algorithm, which works by determining the axes of maximum variance in the collection of observations [74]. Another often-used dimension reduction technique is Multidimensional Scaling (MDS), which computes pairwise distances between observations in the high-dimensional space and attempts to preserve those distances in a low-dimensional projection. MDS implementations exist in both linear and nonlinear forms.

Many of these dimension reduction algorithms require a distance function as input, which provides the method for calculating the similarity of each pair of observations. Much like the breadth of algorithms discussed, a number of distance functions are used in Visualization systems. The most popular metrics are those derived from p-norms, which give distance functions of the form

$$d_p(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_k |x_{i,k} - x_{j,k}|^p \right)^{1/p}.$$

Such a distance is defined for any positive p. The most familiar examples are $p = 1$, which is known as Manhattan distance (due to the city's regular grid structure), and $p = 2$, which is Euclidean distance. Aggarwal et al. [2] showed that Manhattan distances are preferable to Euclidean distances for high-dimensional data, as Euclidean distance (and p-norms of $p > 1$ in general) tends to compress the space as more dimensions are added, resulting in high-dimensional distances that are less distinguishable. In determining the appropriate distance function, it is also worth considering that some distance functions are computationally more difficult when optimizing a stress function.

Large datasets present performance difficulties with some dimension reduction algorithms. For example, MDS requires a distance to be computed between each pair of observations, resulting in $\sim n^2/2$ distances computed for n observations. However, tools do exist to visualize such large datasets. ASK-GraphView supports the interactive visualization of graphs with up to 200,000 nodes and 16,000,000 edges by using clustering algorithms to construct a hierarchical graph, thus visualizing only internal subsections of the graph at any time [1]. A related solution that combines these algorithm families is to initially cluster the data and then apply a dimension reduction algorithm such as MDS on the cluster centroids, followed by subsequent dimension reduction executions on each individual cluster. This minimizes the amount of memory required to store pairwise distances at the expense of no longer having a single global distance measure.

2.2 Clustering Algorithms

Hundreds of clustering algorithms have been implemented, each with inherent strengths and weaknesses. The broad collection of approaches in this class of algorithms stems from the notion that a "cluster" is inherently a subjective structure, and as such cannot be precisely defined. Therefore, new algorithms or improvements on existing algorithms are often created to solve a single problem, though these new solutions may be applied to future problems where appropriate. As a result, there is no globally optimal clustering algorithm; the best clustering algorithm is problem-specific and often determined experimentally [36]. Surveys of clustering algorithms exist in the literature, which include clustering from the perspectives of machine learning, human-computer interaction, visualization, and statistics [23, 92].

Clustering algorithms come in two primary forms: hierarchical and partitioning. Hierarchical algorithms in turn can be divisive (top-down) or agglomerative (bottom-up). The divisive strategy approaches the identification of clusters through iterative partitioning, beginning with a

Table 2. Sample exploratory data analysis tasks, organized by stage in the data analysis process (rows) and algorithm family (columns).

	Dimension Reduction	**Both**	**Clustering**
See the Result	See distribution of observations	See relative positions of observations	Identify clusters of observations
Understand the Result	Measure distances between observations	Identify attribute values of observations	Label clusters Determine cluster structure
Affect the Result	Change distance metric Select different dimensions	Reposition observations in the full space Enhance an existing pattern in the projection	Change cluster membership of observations Create/remove clusters

single group and breaking it down into smaller portions according to an algorithm-specific differencing measure [4]. In contrast, agglomerative algorithms approach clustering through iterating aggregation, beginning with every item in its own group and joining groups together through an algorithm-specific similarity measure [78].

Perhaps the most common clustering algorithm is k-means [64], which partitions a dataset into k clusters according to a distance between each observation and the nearest cluster centroid. Finding an optimal k-means solution is an NP-Hard problem; therefore, heuristic algorithms exist to converge quickly to a local solution. The k-means algorithm has been extended to support a variety of tasks, including weighted clustering [48], hierarchical clustering [75], textual data [26], and constrained clustering [89]. A number of k-means variants are discussed in detail by Cordeiro de Amorim and Mirkin [25]. A major limitation of k-means is that it can only find clusters with convex shapes. The algorithm also requires input parameter k for the number of clusters to create, presenting an additional complication in generating the best set of clusters with its heuristic approach. Several solutions to determine the most appropriate k value are used, such as the elbow method [83].

Many of the distance functions that are applied to dimension reduction algorithms are also useful when considering cluster computations. Cosine distance, for example, can be used to measure cohesion within clusters [81]. The Jaccard similarity coefficient is used for measuring diversity and dissimilarity between clusters or sets of observations [60].

In selecting a clustering algorithm, an additional consideration should be whether an observation can be assigned to only one cluster ("hard" clustering) or can belong to multiple clusters ("fuzzy" or "soft" clustering). The Fuzzy C-means Clustering algorithm [9, 30] is a fuzzy extension of the k-means algorithm, in which the centroid of a cluster is now computed as the mean of all observations weighted by their probability of belonging to the cluster. Fuzzy C-means has found use in the fields of bioinformatics [87] and image analysis [3].

A common clustering tool used in statistics is the Dirichlet process mixture model (DPMM) [10]. This is a probabilistic method, and rather than return a hard clustering assignment, it gives each observation a probability of belonging to any given cluster. Additionally, and unlike k-means, DPMMs learn the number of clusters dynamically, creating new clusters and closing old ones as the algorithm proceeds. It is not without drawbacks, however. The DPMM requires specification of a probability model for the observations in each cluster, which in turn introduces its own difficulties. The algorithm also scales more poorly than k-means with additional data, especially if the model parameters are estimated with Markov chain Monte Carlo.

Much like with dimension reduction algorithms, large datasets can present performance issues with clustering algorithms. Consider again the k-means algorithm, for which the common Lloyd's algorithm heuristic implementation has a running time of $O(nkdi)$ for a dataset with n observations of d dimensions each, k clusters, and i iterations before convergence [64]. The runtime of this algorithm is thus linear in terms of both the number of observations and the number of dimensions; however, performance can be greatly improved by reducing the number of dimensions. Assuming that n and k are fixed, the execution time of the k-means algorithm can hence be improved substantially with dimension reduction, potentially dropping the value of d from hundreds to two (which may simultaneously reduce i as well). Several clustering algorithms are designed to use on large datasets. For one example, the Bradley-Fayyad-Reina (B-F-R) clustering algorithm is a variant of k-means that works by internally maintaining summaries of large collections of observations [13].

3 Tasks

This section presents an overview of tasks commonly seen in visualization systems that implement dimension reduction and clustering algorithms for exploratory data analysis. We discuss the implications of the order in which these algorithms are executed, along with related design decisions and considerations.

3.1 Dimension Reduction and Clustering Tasks

Our discussion of tasks for dimension reduction and clustering algorithms focuses on exploratory data analysis tasks. When exploring a high-dimensional dataset with dimension-reduced projections, there are an immense number of possible 2D- or 3D-projections that can be generated from the dataset. An analyst should be afforded the ability to explore these alternate projections, as well as the related clusterings in those projections, in order to gain insight from the data.

One method for enabling this exploration is by applying weights to the dimensions in the dataset. Biasing the algorithms towards combinations of dimensions in the dataset enables the creation of projections that are similarly biased towards those dimension combinations. Thus, an analyst can explore clusters and patterns in a projection that is biased towards dimensions X, Y, and Z, and contrast that result with clusters and patterns in a projection biased towards only dimensions U and V, both from the same initial high-dimensional dataset.

When interactively exploring a dataset, dimension reduction tasks (the left columns of Table 2) typically relate to position, while clustering tasks (the right columns of Table 2) typically relate to grouping. For example, identifying a similarity relationship between two observations based on their separation distance in a projection is a dimension reduction task, while positioning two similar observations close together is a clustering task. However, there exists obvious ambiguity even with such basic interactions. When positioning two objects close together to form a cluster, the analyst is also communicating a distance relationship between those observations. Thus, space is overloaded for both grouping and layout interactions, further suggesting a relationship between the dimension reduction and clustering algorithm families. As seen in the selected tasks breakdown in Table 2, tasks can often be addressed by only using a dimension reduction algorithm or a clustering algorithm, but there do exist many cases where the interplay between algorithms affects both when a task is performed.

This relationship can be further seen in Brehmer et al. [15], in which ten analysts from six application domains were interviewed with the goal of understanding how analysts explore dimension-reduced data. The end result of this study was a set of five task sequences. Although the authors were focused on analyst interpretations of dimension-reduced data, three of the five resulting task sequences were related to clusters of items revealed in the low-dimensional data projection. Indeed, the "Verify Clusters" task sequence was performed by all ten of their analysts and the "Name Clusters" sequence was performed by eight of the ten analysts. In contrast, the tasks sequences that were not cluster-based were only performed by two ("Name Synthesized Dimensions") and four ("Map Synthesized to Original Dimensions") of the ten analysts. These findings suggest that analysts are discretizing these clusters of observations in dimension-reduced projections. In other words, the dimension reduction algorithm is creating a continuous visual distribution that analysts interpret in discrete segments. Moreover, investigating these clusters within the projection are common goals of user exploration and interaction with datasets.

In addition to investigating clusters in an existing projection, studies have shown that analysts create their own clusters of observations. For

example, the "Space to Think" study by Andrews et al. [8] investigated how analysts use large displays to navigate and lay out documents in the sensemaking process [76], and that these clusters occasionally have spatial relationships, both to develop a timeline and to keep similar clusters of documents near to each other spatially. When interviewed about their sensemaking process later, analysts spoke of their documents and clusters both in terms of proximity and in terms of groups, implying that these are similar cognitive processes. The ForceSPIRE [33] and StarSPIRE [12] systems were designed in part from these findings.

Similar behavior was seen in the "Be the Data" system reported by Chen et al. [20], which allows participants to explore a dataset by taking on the role of the observations in a defined physical space. By moving about the space, participants update a dimension-reduced projection. The system is thereby able to learn which dimensions of the dataset are most important to the current "projection" of people. Presented with a collection of animals and their attributes, a group of seventh grade students were posed the question "What makes some animals good to eat?" The students began their exploration of the data by clustering animals into discrete Edible and Inedible clusters. However, the student who embodied the Rat observation did not consider herself a part of either group, noting that rats are normally not edible but are consumed in some cultures. She then positioned herself between the two clusters. This caused the rest of the students to reconsider their distribution, turning the discrete clusters into a continuous distribution of Edibility.

Cluster investigation tasks (the right columns of Table 2) come in a number of forms, each of which have some meaning in a dimension-reduced projection. For example, analysts may wish to understand the overall layout of clusters in a projection, explore the proximity of one cluster to another, investigate clusters of clusters and similar structures, the shape of a cluster, and describe outlying clusters versus central clusters. In addition, analysts may be interested in the relationship between clusters and the individual observations in the projection, exploring to which cluster(s) an observation belongs, understanding the properties of observations that are outliers to all clusters, and investigating the properties of a set of observations that form a cluster. There is a mix of distribution and group questions that can be addressed through the combination of both dimension reduction and clustering algorithms.

Adding clusters and clustering interactions to dimension-reduced data can also improve scalability as datasets continue to grow in size [32]. Having the ability to abstract collections of observations into a single cluster that acts as an interaction target enables the ability to place more objects into virtual spaces, useful both for standard monitors and for large display systems.

While the outputs of dimension reduction and clustering algorithms are useful to locate patterns in a dataset, we also benefit from enabling these algorithms to learn from user interactions [34, 46, 59]. By interpreting the semantic meaning of user interactions, each of these algorithms can better enable exploratory data analysis. For example, an analyst may wish to know what model parameters are necessary to create a cluster from observations A, B, and C. By manipulating the projection to form such a cluster and initiating a semi-supervised machine learning routine, the dimension reduction and clustering algorithms can be trained to learn such model parameters and to update the entire projection in response to those new parameters. The new projection may create a new cluster from observations D, E, and F in addition to the analyst-created cluster, a new insight into the dataset. Therefore, the dimension reduction and clustering algorithms can help both at the beginning of the exploration process by providing a naïve starting point, as well as throughout the exploration process by responding to the interactions of an analyst.

3.2 Coordinating the Algorithms

Another consideration in selecting dimension reduction and clustering algorithms is determining what parameters should be learned and used by each algorithm, as well as what information should be learned by the analyst. Beginning with the analyst, we discussed in the previous subsection that dimension reduction algorithms and clustering algorithms serve similar purposes. However, dimension reduction algorithms are more suited to tasks for pairwise comparisons and similarities between observations, while clustering algorithms are better suited for comparisons involving the recognition and description of groups.

For the algorithms, one obvious design decision is to determine whether or not the dimension reduction algorithm and clustering algorithm should be using the same distance function, or even if they should be using the same set of weights on the dimensions. It is possible for the dimension reduction algorithm and the clustering algorithm to store separate sets of weights, or to use different distance functions entirely.

When considering the semantics of the order of dimension reduction and clustering algorithms, using clustering in high-dimensional space as the first operation makes uncovering clusters the primary semantic role of the system, and hence results in a system designed to support locating and understanding groups in the input data. In contrast, clustering as the second operation in the low-dimensional space after executing a dimensional reduction algorithm results in a clustering algorithm that is merely a secondary aid to the dimension reduction algorithm.

An open question is determining whether analysts are cognitively clustering in high-dimensional or low-dimensional space. Given that analysts typically form clusters of text documents directly from the text instead of first converting those documents into another form [8], it appears that clustering is performed in the high-dimensional space, at least for textual data. Understanding the clustering process of analysts will lead to better semantic interactions in this dimension reduction and clustering design space, leading to further system interactions such as enabling humans to provide corrections to clustering assignments and hence update dimension reduction algorithm weights and projections.

Naturally, it is not possible to coordinate all pairs of dimension reduction and clustering algorithms. For example, some dimension reduction algorithms such as PCA do not rely on distances between observations. Therefore, using the same distance measure between PCA and a clustering algorithm is not possible.

3.3 Dimension Reduction and Clustering Combinations

When developing a system that includes both dimension reduction and clustering algorithms, it is important to consider the order in which these algorithms are performed on the data, as the order of these algorithms will generate projections with different semantic meanings. Fig. 2 includes six different pipelines that display execution orders and data flows between these algorithms. Each of these pipelines is discussed in the following paragraphs. As the analyst progressively explores the dataset, they may select a different pipeline for each round of exploration, continuing to explore new projections (Fig. 3).

Independent Algorithms: As discussed previously, many visualization systems incorporate both dimension reduction and clustering algorithms, but these algorithms often execute independently and in parallel so that the output of one algorithm has no effect on the other. This pipeline is highlighted first in Fig. 2 and was discussed in the iVisClustering [57] example in the Introduction. In this system, topics are computed and assigned as clusters, and a force-directed computation performs the node layout in the spatialization. However, an update to the layout has no effect on the clustering assignments. In addition, computing both algorithms on the high-dimensional data will be more computationally expensive than performing only a single high-dimensional computation.

Dimension Reduction Preprocessing for Clustering: Another possibility is to execute a dimension reduction algorithm on the high-dimensional data, and then pass the low-dimensional projection to the clustering algorithm to determine groups, clustering on the reduced data rather than the source data. This decision may be advantageous because the clustering algorithm can execute faster on a dataset with fewer dimensions, but the outcome may be misleading because the low-dimensional positions of each observation are an approximation of the high-dimensional relationships. Rather than generating clusters of the input data, we generate clusters using data with less information, resulting in potentially misleading cluster assignments. This risk is discussed by Joia et al. [52], noting that distances in the low-dimensional space may be misleading due to projection errors. As a result, what appear to be distinct clusters must be confirmed, as there is no guarantee that these clusters do contain unique content. An example of this

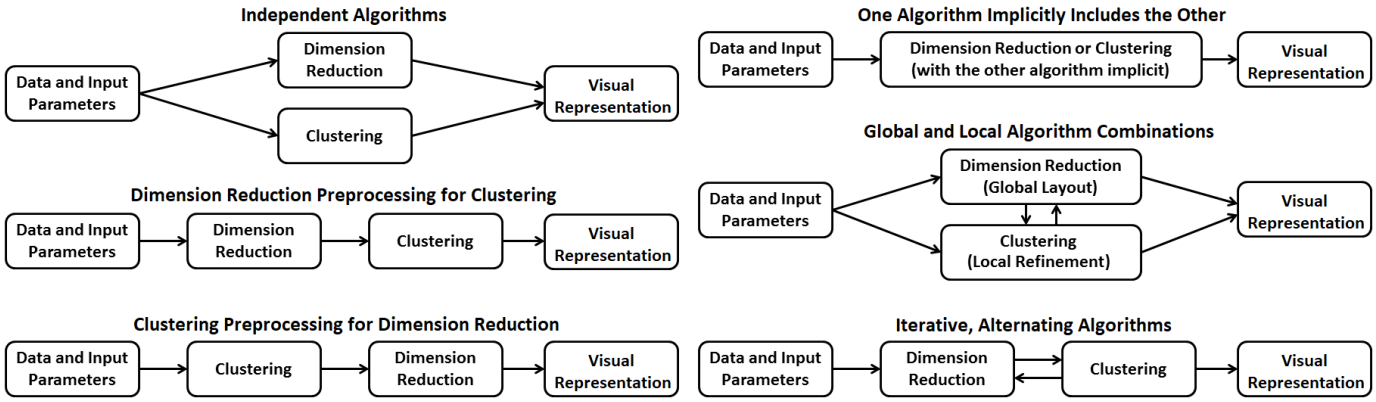


Fig. 2. Six different options for pipelines depicting combinations of dimension reduction algorithms and clustering algorithms. In each of these pipelines examples, it is implied that each algorithm could use an independent distance function, resulting in more than just these six pipelines. Further, these pipelines represent a single analysis iteration.

pipeline can be seen in Zha et al. [94], in which a technique similar to PCA is performed first and followed by k-means on that output. Likewise, Ng et al. [71] propose an algorithm in which the observations are embedded in low-dimensional space such as the eigenspace of the graph Laplacian, and then k-means is applied to that low-dimensional projection. Be The Data also creates clusters dynamically based on the current projection [20].

Clustering Preprocessing for Dimension Reduction: The reverse of the previous behavior occurs when the clustering algorithm is the first to execute, and then some information from the clustering output (the cluster assignments, or the locations of the centroids) is used by the dimension reduction algorithm for layout. Now, the clusters represent relationships that exist in the initial data in high-dimensional space. However, the clustering algorithm will take longer to execute due to the additional number of dimensions processed. While less common, some systems do operate in this way. For example, Ding and Li first use k-means clustering to initially generate class labels, followed by LDA dimension reduction for subspace selection [28]. Fuzzy clustering introduces a new complexity to this pipeline, as cluster assignments are now a probability distribution rather than a fixed bin assignment. A pipeline of this form can also improve scalability, as the time and space complexity of many dimension reduction algorithms make them infeasible to execute on very large datasets. Clustering observations and then performing a dimension reduction algorithm on those clusters is one solution to this challenge.

One Algorithm Implicitly Includes the Other: Another alternative is to only execute one of the algorithms, either dimension reduction or clustering, and then convert or interpret the output of the executed algorithm as the output for the other algorithm as well. In these cases, the results from one algorithm are structured to fit the objective of the other algorithm, exploiting the mathematical equivalence between these algorithm families discussed briefly in the Introduction. For example, we can codify soft k-means clustering as assigning n observations to k features with some associated weight or probability. Likewise, we can formulate dimension reduction as reducing m features to p features with some associated weight or probability. Therefore, the outcome of soft k-means clustering can be interpreted in terms of dimension reduction by making the k clustering features also represent the p dimension reduction features. A similar argument exists to map the outcome of a dimension reduction algorithm directly to a cluster encoding by executing a dimension reduction algorithm like PCA and binning the output along one of the axes. Perhaps a more straightforward example of this pipeline is the self-organizing map [55], a dimension reduction technique which can be directly interpreted as a set of clusters without any feature transformation. Kriegal et al. [56] present a survey of clustering techniques for high-dimensional data, and include a discussion on subspace clustering algorithms. Such algorithms simultaneously reduce both the number of observations and the number of dimensions in a dataset, in contrast with having a dimension reduction algorithm that reduces the number of dimensions computing separately from a clustering algorithm that reduces the number of observations.

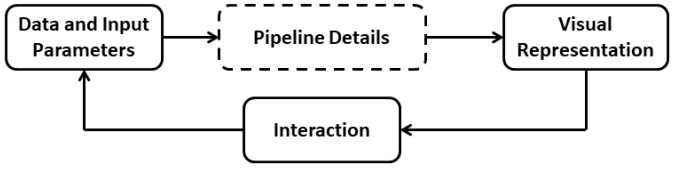
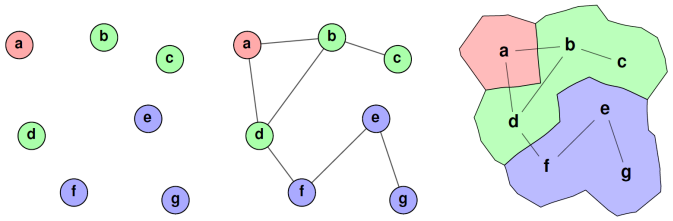


Fig. 3. Interactions from the analyst will drive additional executions through the pipeline during the data exploration process. The analyst does not need to select the same pipeline on every iteration of the analysis.

Global and Local Algorithm Combinations: Because dimension reduction algorithms typically take a global view of the overall space while clustering algorithms take a local view [26], another option is to implement a pipeline in which the overall structure of the space is informed by the dimension reduction algorithm while local structures are governed by the clustering algorithm. These algorithms can communicate with each other to converge towards an optimal layout, but each is responsible for its own aspect of the structure. To further clarify the difference between this pipeline and some of those discussed previously, consider organizing a large collection of documents in a display. One possibility is to place related documents into folders, and then organize the folders in the space. This example reflects the "Clustering Preprocessing" pipeline, as we organize the clusters rather than individual documents. In contrast, the analyst could organize groups of documents in the space, and then select and move those groups with respect to one another. This example affords some additional fuzzy clustering capabilities, as a document that may belong to two or more clusters can be placed between those clusters. Here, the overall layout of the documents can be handled by dimension reduction, while some local structures of similar documents are supported by clustering.

Iterative, Alternating Algorithms: The final pipeline represents a structure where both dimension reduction and clustering are working together in the same overarching algorithm. As k-means is an algorithm that alternates between updating cluster assignments and centroid positions, a third stage can be added for dimension reduction. Ideally, this iterative alternating process will enable dimension reduction and clustering to work in harmony to converge towards a best layout, trying to find the right set of dimensions and a good set of clusters simultaneously while also communicating between the algorithms. This pipeline differs from "One Algorithm Implicitly Includes the Other" in that both algorithms process the data cooperatively, rather than only executing one of the algorithms and using its outcome to present both a projection



(a) N (b) NL (c) NLG

Fig. 4. Three options for encoding group membership as studied by Saket et al [79]. In (a), nodes are free-floating and colored based on cluster membership. In (b), the cluster coloring remains, and links are drawn as necessary between some of the nodes. In (c), the nodes are replaced by colored space-filling regions to indicate cluster membership.

and a clustering. Since both the dimension reduction algorithm and the clustering algorithm will begin on the high-dimensional data, this pipeline will be among the slowest to converge. Niu et al. [72] provides an example of this pipeline.

This collection of pipelines and examples demonstrates methods for combining dimension reduction and clustering algorithms, but are not without limitations. Even extending these pipelines with a looping structure to iterate through the dimension reduction and clustering stages is insufficient. To better model this and other similar cognitive processes, we must extend this discussion of algorithms into the realm of visualization and interaction; algorithms alone are insufficient for complex cognition [35].

4 Visual Representation

After the algorithms have been selected, the next step is determining how to present the results of the computations to the analyst. In this section, we first discuss common visual representations for dimension-reduced data and clustered data. This is followed by a discussion of potential visual outcomes of the pipelines introduced in Sect. 3.3.

4.1 Known Visualization Issues

As the sample interfaces in the bottom row of Fig. 6 show, most dimension reduction algorithm outputs are shown in scatterplots or node-link diagrams. These scatterplots come with inherent issues in some cases, such as difficulties in displaying and interpreting the dimensions that result from an MDS projection. When dealing with large datasets, the scatterplot or node-link representation of the dimension reduction output runs a high risk of overplotting, especially if the spatialization exhibits clear clustering in the layout. One solution for overplotting is to abstract a cluster of observations into a single glyph to represent a collection of observations, such as suggested by the Splatterplots implementation [67]. An alternative is to filter the number of observations visible in an overdrawn region, keeping a representative ratio of each cluster in the overdrawn region [19].

While the natural representation of the dimension reduction output uses a spatial projection like a scatterplot or node-link diagram, the possibilities for representing cluster membership are much more diverse. In addition to demonstrating clusters using a collection of nodes in close spatial proximity, cluster membership can be encoded with colors or glyphs. Even then, a number of design decisions can be made for how best to express these memberships by color and shape.

Saket et al. [79] evaluate several encodings of cluster information (see Fig. 4 for a visual representation of each of these encodings), relating each to node-based tasks (for example, "Given node X, what is its background color?") and group-based tasks ("Given nodes X and Y, determine if they belong to the same group"). They found that the addition of group encodings does not negatively impact time and accuracy on node-based tasks. As would be expected, group-based tasks were best solved by node-link-group encodings. This outcome suggests that the visual representation used to encode the clusters in the projection depends on the tasks that the system addresses.

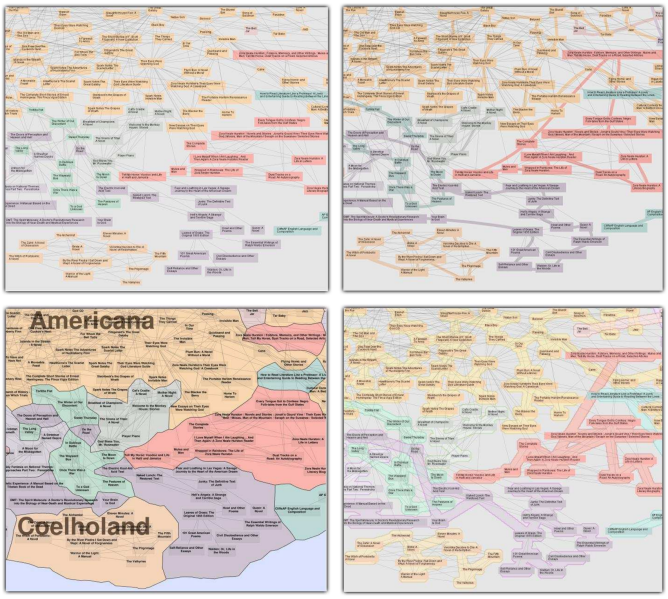


Fig. 5. Four options for displaying cluster membership as studied by Jianu et al [51]. In addition to a node-link representation similar to that included by Saket et al., this study included Linesets [5], GMap [41], and BubbleSets [24].

Jianu et al. [51] perform a similar evaluation on four visual representations, including a node-link diagram similar to that studied by Saket et al. as well as three other visual representations that are shown in Fig. 5. Linesets [5] include link colors that match the node colors representing cluster membership, highlighting connections between nodes that are in the same cluster or group (top-right of Fig. 5). GMap [41] is a space-filling representation that renders a geographic-like map for clusters, containing all of the nodes in a colored region similar to the node-link-graph representation studied by Saket et al. (bottom-left of Fig. 5). Finally, BubbleSets [24] draws isocontours around clusters, effectively balancing the Linesets and GMap representations by using the isocontours to highlight links connecting members of the same cluster but becoming space-filling in regions with high node density (bottom-right of Fig. 5). This study found that BubbleSets was the superior representation for group-based tasks, but that encoding group information onto node-link diagrams adds a 25% time penalty onto network-based tasks, a conflict with the conclusion of Saket et al. Clearly, more research is needed in this area to resolve such conflicts.

In addition to the above, another method for visualizing clusters in a scatterplot or node-link diagram is to enclose nodes from individual clusters in a convex hull [90]. Because k-means solves for convex clusters based on a distance from an observation to the nearest cluster centroid, a convex hull visualization may be the most natural visualization representation for a k-means clustering output.

Moving away from scatterplot and node-link representations, an alternative representation for clusters is to encode topics into a streamgraph. For example, Liu et al. use streamgraphs to encode related text keywords into topical collections, using the streamgraph to show how the importance of those topics and keywords changes over time [63].

4.2 Algorithm Order Visualizations

Designers have an additional choice regarding which features are emphasized in the visual representation. For example, should the spatial layout of the dimension reduction be emphasized over the cluster assignments? Alternatively, should the cluster assignments inform the layout of the observations? Should we attempt to balance the two outputs? How much of an impact should the algorithm order play in the final layout? The order in which we execute the dimension reduction and clustering algorithms should have some impact on the outcome of the visualization, but the degree to which this execution order is

emphasized can vary by system goals. Here, we describe potential visualization properties for each of the pipelines described in Sect. 3.3.

Independent Algorithms: Consider the first pipeline from Fig. 2, in which both algorithms execute independently and in parallel. One potential outcome of this pipeline is to represent clusters using convex hulls. Here, the dimension reduction algorithm operates to find an ideal layout, while the clustering algorithm separately finds an ideal cluster set. When combining the outputs, a potential result is a cluttered visualization that is somewhat ambiguous in the cluster assignments of some observations due to intersections between the clusters. A potentially better solution, used by iVisClustering [57], is to use nodes colored by class in cases of cluster occlusion such as these. Another solution that allows the convex hulls to remain is to implement layout constraints (such as those in IPSep-CoLa [31]) so that objects that clearly belong to different clusters are visibly separated in the spatialization. However, this requires prior knowledge of key cluster-defining objects, or an initial clustering computation that precedes the main clustering process. This also defeats the goal of the pipeline by removing the separation between dimension reduction and clustering algorithm execution.

Dimension Reduction Preprocessing for Clustering: In this pipeline, the output of the dimension reduction algorithm is fed into the clustering algorithm, enabling clustering on the low-dimensional reduced data rather than on the initial high-dimensional data. Because clusters are drawn based on the proximity of observations in the projection, it is unlikely that these clusters will intersect. As noted previously, executing the clustering algorithm on the dimension-reduced data may not produce an optimal clustering on the high-dimensional data, which could affect the analyst's comprehension of the projection.

Clustering Preprocessing for Dimension Reduction: In the reverse of this process, we now cluster in the initial high-dimensional data, and use some of that information such as the cluster assignments to inform the dimension reduction. Such a visualization will likely result in visibly separated clusters as in the previous case, though perhaps even more separated because space can be artificially added between the clusters. Again, because we execute the dimension reduction algorithm on the cluster assignment information (or other cluster algorithm output) rather than on the initial high-dimensional data, the dimension reduction projection may not be optimal and could also affect the analyst's comprehension of the projection. More clearly stated, two points that the dimension reduction algorithm judges to be somewhat similar (but not similar enough to belong to the same cluster) may have an artificially large distance applied between them in this projection.

One Algorithm Implicitly Includes the Other: A pipeline in which only one algorithm is executed to perform both the dimension reduction and clustering functions has inherent limitations depending on which algorithm is performed. For example, if the dimension reduction algorithm is executed and clustering is applied only on the result of the dimension-reduced spatialization, the clustering will likely be far from optimal but the dimension reduction will be ideal. This could result in a visualization in which, for example, the clusters are simply assigned based on x-position in the projection.

Global and Local Algorithm Combinations: The global and local pipeline describes the dimension reduction algorithm as responsible for the global layout, while the clustering algorithm is responsible for local refinements and layout. These algorithms work together to create an overall layout in which the dimension reduction algorithm effectively lays out the clusters in a meaningful manner while the internal structure of each cluster is maintained by the clustering algorithm. As such, the fine details of the projection will not be as accurate spatially as the dimension reduction outcomes in the Independent Algorithms and Dimension Reduction Preprocessing for Clustering pipelines, and the clustering is still executing in part on the low-dimensional projection. However, the layout should be relatively clean and understandable, and the overall structure of the projection (e.g., the relative positions of the clusters) will be meaningful.

Iterative, Alternating Algorithms: The final pipeline in Fig. 2 includes both the dimension reduction algorithm and the clustering algorithm working simultaneously and collaboratively to structure a projection that is near-optimal for both representations. As such, this

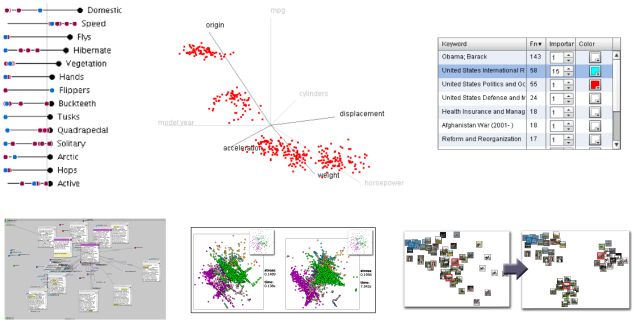


Fig. 6. A selection of interfaces and tools that support Parametric Interaction or Observation-Level Interaction. The upper row shows PI interfaces that include slider bars from Andromeda (PI view) [80], Star Coordinates [53], and SpinBox widgets from STREAMIT [6]. The lower row shows OLI interfaces from StarSPIRE [12], Paulovich et al. [73], and Mamani et al. [66].

structure may produce the best visualizations with respect to the meaning of the data, albeit at the cost of runtime.

A number of further design decisions can be incorporated into the visualization. We have the option to emphasize the relative distance between clusters more than the relative distance between pairs of observations. The visualization space is thus clusters of observations that are obviously separated from each other in the space, possibly with another iteration of the dimension reduction algorithm performed on each individual cluster to generate a local layout. As yet another alternative, if the analyst is most interested in the clusters in the projection, the emphasis could also be placed on the distance between each observation and the centroid of the cluster that it belongs to. Clusters could also be artificially separated by a secondary execution of the dimension reduction algorithm, but the superior layout determination is dependent on the distance between each observation and a centroid.

We noted in Sect. 3.2 that it is not possible to combine all pairs of dimension reduction and clustering algorithms. Likewise, it is not possible to include all visual representations of dimension reduction and clustering in the same visualization. For example, dendrograms are often used to show hierarchical clustering; however, dendrograms are not a useful visual encoding for dimension reduction algorithms.

5 Interacting with Projections and Clusters

After displaying a visualization of dimension-reduced and clustered data, the next step is to provide interactions to afford user exploration through the dataset. Many studies have been performed and taxonomies generated for interacting with high-dimensional data in a data analytics context [7, 17, 88, 93].

In the context of exploring dimension-reduced data projections, two primary methods exist for modifying an underlying distance function: Parametric Interaction and Observation-Level Interaction. Surface-level interactions are also often incorporated into visualization systems, though these do not modify the underlying model. We begin this section by discussing these interaction techniques and some representative tools, as well as discussing interaction techniques that address clustering challenges. We follow this with a discussion of potential interaction techniques that can support interaction with both dimension reduction and clustering algorithms simultaneously.

5.1 Current Interaction Techniques

Parametric Interaction (PI) refers to manipulating parameters directly in order to create a new projection and/or clustering assignment. This presents a difficulty to novice or non-mathematically-inclined analysts, who may not understand how to update a set of weights to create the dimension-reduced projection that they desire. In contrast, Observation-Level Interaction (OLI) refers to direct manipulation of the observations, which in turn triggers a backsolving routine to learn new parameters [34, 46, 59]. In this way, OLI hides the manipulation

Table 3. Sample interactions, organized by type of interaction (rows) and by the type of algorithm affected by the interaction (columns).

	Dimension Reduction	Both	Clustering
PI	Rotate the projection	Modify the weight on a dimension Select a different distance function	Modify the max/min radius of a cluster Change the number of clusters sought
OLI	Reposition an observation external to clusters or within a single cluster	Reposition an observation into a different cluster	Change cluster membership Merge several clusters or split a cluster
Surface	Measure a distance between observations	Details-on-demand to obtain attribute values	Count the size of a cluster Annotate a cluster

of the model from the analyst, allowing the analyst to perform more natural direct manipulation interactions with the observations themselves. In Andromeda [80], PI allows analysts to modify weights on the dimensions to modify the distance function directly by interacting with sliders, while OLI uses an inverse MDS computation to interpret the semantic meaning of the interaction in order to solve for those weights.

The upper row of Fig. 6 shows sample examples of PI from recent visualization systems, complemented by some representative interactions in the upper row of Table 3. Horizontal and vertical slider bars are frequently utilized to enable analysts to interact with model parameters, despite the fact that these model parameters have a variety of contexts. Some of these sliders, such as those in Andromeda [80], include additional glyphs on the sliders to show the values of selected observations on each dimension. In addition to slider bars, other techniques have been utilized to support the manipulation of model parameters, such as the SpinBox widgets of STREAMIT [6] and the transforming axes of Star Coordinates [53]. PI techniques can also be extended to interact with dimensions as well as observations, as shown by Turkay et al [86].

As seen in the lower row of Fig. 6 and discussed in Sect. 4, scatter plots and node-link diagrams are the overwhelming favorite for displaying dimension-reduced projections, including those that support OLI. Despite the ubiquity of these visual representations, individual OLI systems do display unique features and properties, such as supplementing the scatterplot with additional views for context [16], supporting PI in addition to OLI on the scatterplot [80], including local transformations [66], and focusing exclusively on textual data [12].

An additional consideration for OLI is the "With Respect to What" problem detailed by Self et al. [80], which is the fundamental challenge of using rigid algorithms to interpret the ambiguous meaning of an interaction that involves dragging a node from one part of the display to another. Andromeda solves this challenge by defining a radius at both the starting and ending point of the interaction, implying that the analyst is moving an observation away from all other observations within x pixels of the source and towards all other observations within x pixels of the destination of the interaction, though the analyst is afforded the ability to deselect observations that do not apply to the interaction [80]. Points contained within this radius are highlighted in the visual representation, allowing analysts to clearly see the interaction targets that they are expressing within the projection [47].

In addition to Parametric and Observation-Level Interactions, the introduction of clusters affords a variety of cluster-based interactions that can support sensemaking. To begin, OLI can be applied to clusters, including such interactions as moving clusters together and further apart to reflect similarities and differences between clusters, as well as transferring that information either to the weights on the clusters or the weights on the nodes. We can also apply parameter tuning to clusters at a global level, changing the number of clusters or the radius of all clusters, or we can tune the parameters of individual clusters, creating a collection of clusters with a variety of radii. The Vizster system, for example, includes a PI-style slider bar to change the number of clusters displayed in the X-ray view [44].

Clusters also introduce new cluster-specific interactions, such as cluster merging, splitting, and creation [22,45], cluster annotation [54], and hierarchies of clusters [70]. Performing any of these interactions can communicate semantic information back to the system, re-executing the pipeline that may or may not also include re-executing the dimension reduction algorithm as a result of this user interaction.

5.2 Combined Interaction Techniques

The pipelines discussed in Sect. 3.3 naturally support the Parametric, Observation-Level, surface-level, and clustering interactions discussed in the previous subsection. Interactions in general can be designed for each of these pipelines individually, but it is also useful to consider interactions that can have meaning to both the dimension reduction algorithm and the clustering algorithm simultaneously. To do so means facing similar ambiguity that is addressed by the "With Respect to What" problem and the issue of overloaded space.

For example, consider an analyst who is interacting with the clustering assignment in a projection. Regardless of whether the analyst is interacting with high-dimensional or low-dimensional clusters, dragging an observation from one cluster to another is a natural interaction to correct a misclassification. However, the cause of that misclassification may be unknown to the analyst. Perhaps the analyst is interacting with a system that implements the dimension reduction preprocessing pipeline. If that is the case, then the analyst may be correcting a misclassification that results from the clustering operating on the projected low-dimensional data. Thus, the goal of the system should be to learn from that interaction, with the goal of getting closer to the ideal high-dimensional clustering.

Alternatively, if the analyst is interacting with a system that implements clustering on the high-dimensional data, then performing the same interaction is correcting for a case where the heuristic clustering algorithm did not find the optimal solution. The system can still learn from this interaction to correct future clusterings, but the different cause of the misclassification should result in a different model update. These two misclassification corrections may be semantically identical to the analyst who seeks to correct an error, but the underlying mechanics that caused and must correct the misclassification are different.

The same is true of an analyst interacting with observations in a dimension-reduced projection. If an analyst drags an observation, it may simply be that the analyst wishes to adjust the strength of the relationship between two observations. However, adjusting the strength of a relationship calculated on the high-dimensional data is inherently different than adjusting the strength of a relationship calculated on cluster algorithm output. And does the semantic meaning of the interaction change if that drag interaction crosses a cluster boundary?

The introduction of explicitly-defined clusters allows for a formal target against which to judge interactions. When explicit clusters are defined, the analyst has four clearly defined "with respect to what" operations: (1) moving an observation into a cluster, (2) moving an observation out of a cluster, (3) moving an observation from one cluster into another, and (4) moving an observation without changing cluster membership [90]. Each of these interactions can be designed to have an effect on both the dimension reduction algorithm and the clustering algorithm. Keeping an observation within a cluster, or dragging it from one cluster into another, provides information to the clustering algorithm that the classification is either correct or incorrect. At the same time, relocating an observation to a different position communicates suggested distance information between the moved observation and one or more additional observations in the projection. Each of these algorithms can thus work to update the weight vector that then leads to a projection and clustering update with this new information.

When mapping interactions to the pipelines summarized in Fig. 2, choosing the primary target of the interaction is important even when an interaction affects both algorithms. In the previous example, the

pipeline is implemented with the interaction primarily occurring on the clusters, changing the cluster assignment of observations in order to update the dimension reduction projection [90]. In contrast, "Be the Data" also implements the same pipeline but with an interaction primarily on the observation layout, using the dimension reduction algorithm to update the clusters [20]. These two systems are both implementations of the same pipeline, but place the interaction on different algorithms to answer different questions about the high-dimensional data. Thus, interactions can be considered independent of the pipelines.

A further open question to be addressed regards interactions on the clusters themselves. If an analyst drags a cluster or interacts with it in another manner, what adjustments should be made to the observations and relationships within that cluster, as well as the relationships that cross that cluster boundary?

6 Discussion

Combining dimension reduction and clustering algorithms into the same visualization system provides a number of opportunities for visualization and interaction design. A system in which the two algorithm classes cooperate for exploratory data analysis results in a relationship in which the projection space (the outcome of the dimension reduction algorithm) helps to explain the meaning of the clusters in the space, while the clusters themselves help to explain the meaning of the space.

Including a machine learning aspect into a visualization system to permit the dimension reduction and clustering algorithms to learn from the actions of the analyst presents a number of additional challenges for interaction design. In particular, the overloaded space metaphor discussed in Sect. 3.1 causes challenges, as interactions within the system must be mapped to at least one algorithm and may ambiguously be mapped to both. For example, if an analyst drags and drops a datapoint to reposition it in space, but the new coordinates did not result in a cluster reassignment, should the clustering algorithm learn nothing, or did the analyst provide some "fuzzy" clustering feedback to the algorithm? A notion of iterative refinement, in which the analyst gradually trains the algorithms and offers corrections to mistakes at each iteration is necessary in these cases. Such an iterative refinement process mimics Pirolli and Card's Sensemaking Process [76].

Maintaining an analyst's mental map during layout adjustments is a well-studied problem [68], and is another factor that should be considered in visualization and interaction design for dimension reduction and clustering systems. ForceSPIRE and Andromeda approach this mental map challenge in different ways. ForceSPIRE, using a force-directed layout, maintains the positions of nearly all observations during an interaction, only altering the positions of observations near the interaction [33]. In Andromeda, on the other hand, it is possible that all observations could move the entire distance across the space. The system cognitively aids the analyst to understand such broad changes with an animation slider, affording the analyst with the ability to incrementally follow the post-interaction transition, as well as a layout stabilization module to suppress the rotation invariant property of the Weighted Multidimensional Scaling dimension reduction algorithm [80].

This work focuses on exploring the breadth of design options available to visualization researchers when combining dimension reduction and clustering algorithms. Our goal with this work is to highlight many of the decisions that exist in this design space, spurring further exploration of this space with new tools. While we present a number of design questions that must be addressed in creating such a visualization system, we do not claim to answer any of these questions, as the answers to many of them depend on the tasks and goals of the system.

7 Conclusion

The combination of dimension reduction and clustering algorithms represents an immense design space, including considerations of algorithm selection and order, tasks, visualization, and interaction. In this paper, we have provided a survey of each of these considerations, describing existing research and discussing relevant design decisions applicable to current and future systems (summarized in Table 4).

Returning to our discussion of the "Be the Data" interaction first addressed in Sect 3, we saw a smooth transition from discrete to continuous thinking. The students initially formed the clusters of Edible and Inedible animals and then positioned those clusters in space, initially mimicking the cluster preprocessing pipeline. The transition from this projection into a spectrum of Edibility amounts to iterative and interactive refinement of those initial clusters into a broader projection. Without the interaction component, the pipelines could not successfully model this student behavior.

A useful future direction for research would be a cognitive study, further attempting to understand how analysts cognitively combine the ideas of dimension reduction and clustering in both virtual and non-virtual spaces. Such a study can further inform the pipelines, visualizations, and interactions presented in this work.

Table 4. A summary of the design challenges and questions discussed throughout the paper regarding the combination of dimension reduction and clustering algorithms.

Section	Design Decision
2.1 & 2.2	In general, clustering places an emphasis on relationships within and between clusters. In contrast, dimension reduction emphasizes observation-to-observation relationships. Which of these tasks is the primary goal of the analyst?
3.2	What properties of the data is the visualization seeking to highlight? Which properties of the data are the system and analyst trying to discover? Should the primary goal of the visualization system be emphasizing observation relationships, clusters of observations, or both? Should the dimension reduction and clustering algorithms use the same distance function (if possible), or should each algorithm use an independent method of measuring similarity?
3.3	Which order and interaction of dimension reduction and clustering algorithms best models the task that the visualization system is addressing?
4.1	How can we encode distances and cluster membership information when both algorithms are present?
4.2	As the dimension reduction and clustering algorithms are competing in the same visualization, what features should be emphasized in the visualization to best address the problem?
5.2	Should interactions be designed independently for the dimension reduction and clustering algorithms, or should a given interaction affect both algorithms?

Acknowledgments

This research was supported by NSF Grants IIS-1447416, IIS-1633363, and DGE-1545362, as well as by a grant from General Dynamics Mission Systems. The authors would like to recognize the role of comments from reviewers and discussions with InfoVis Lab @ VT research group members in improving this work.

References

[1] J. Abello, F. V. Ham, and N. Krishnan. Ask-graphview: A large scale graph visualization system. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):669–676, Sept 2006. doi: 10.1109/TVCG.2006.120

[2] C. C. Aggarwal, A. Hinneburg, and D. A. Keim. *On the Surprising Behavior of Distance Metrics in High Dimensional Space*, pp. 420–434. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. doi: 10.1007/3-540-44503-X_27

[3] M. N. Ahmed, S. M. Yamany, N. Mohamed, A. A. Farag, and T. Moriarty. A modified fuzzy c-means algorithm for bias field estimation and segmentation of mri data. *IEEE Transactions on Medical Imaging*, 21(3):193–199, March 2002. doi: 10.1109/42.996338

[4] M. S. Aldenderfer and R. K. Blashfield. *Cluster analysis*. SAGE publications, Beverly Hills, USA, 1984.

[5] B. Alper, N. Riche, G. Ramos, and M. Czerwinski. Design study of linesets, a novel set visualization technique. *IEEE Transactions on Visualization*

and Computer Graphics, 17(12):2259–2267, Dec 2011. doi: 10.1109/TVCG.2011.186

[6] J. Alsakran, Y. Chen, Y. Zhao, J. Yang, and D. Luo. Streamit: Dynamic visualization and interactive exploration of text streams. In *2011 IEEE Pacific Visualization Symposium*, pp. 131–138, March 2011. doi: 10.1109/PACIFICVIS.2011.5742382

[7] R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pp. 111–117, Oct 2005. doi: 10.1109/INFVIS.2005.1532136

[8] C. Andrews, A. Endert, and C. North. Space to think: Large high-resolution displays for sensemaking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pp. 55–64. ACM, New York, NY, USA, 2010. doi: 10.1145/1753326.1753336

[9] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1981.

[10] D. M. Blei and M. I. Jordan. Variational inference for dirichlet process mixtures. *Bayesian Analysis*, 1:121–144, 2005.

[11] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[12] L. Bradel, C. North, L. House, and S. Leman. Multi-model semantic interaction for text analytics. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 163–172, Oct 2014. doi: 10.1109/VAST.2014.7042492

[13] P. S. Bradley, U. Fayyad, and C. Reina. Scaling clustering algorithms to large databases. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, KDD'98, pp. 9–15. AAAI Press, 1998.

[14] M. Brehmer and T. Munzner. A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2376–2385, Dec 2013. doi: 10.1109/TVCG.2013.124

[15] M. Brehmer, M. Sedlmair, S. Ingram, and T. Munzner. Visualizing dimensionally-reduced data: Interviews with analysts and a characterization of task sequences. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, BELIV '14, pp. 1–8. ACM, New York, NY, USA, 2014. doi: 10.1145/2669557.2669559

[16] E. T. Brown, J. Liu, C. E. Brodley, and R. Chang. Dis-function: Learning distance functions interactively. In *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 83–92, Oct 2012. doi: 10.1109/VAST.2012.6400486

[17] A. Buja, D. Cook, and D. F. Swayne. Interactive high-dimensional data visualization. *Journal of Computational and Graphical Statistics*, 5(1):78–99, 1996. doi: 10.1080/10618600.1996.10474696

[18] J. D. Carroll and J.-J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[19] H. Chen, W. Chen, H. Mei, Z. Liu, K. Zhou, W. Chen, W. Gu, and K. L. Ma. Visual abstraction and exploration of multi-class scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1683–1692, Dec 2014. doi: 10.1109/TVCG.2014.2346594

[20] X. Chen, J. Z. Self, L. House, and C. North. Be the data: A new approach for immersive analytics. In *IEEE Virtual Reality 2016 Workshop on Immersive Analytics*, 03/2016.

[21] J. Choo, S. Bohn, and H. Park. Two-stage framework for visualization of clustered high dimensional data. In *2009 IEEE Symposium on Visual Analytics Science and Technology*, pp. 67–74, Oct 2009. doi: 10.1109/VAST.2009.5332629

[22] J. Choo, C. Lee, C. K. Reddy, and H. Park. Utopian: User-driven topic modeling based on interactive nonnegative matrix factorization. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):1992–2001, Dec 2013. doi: 10.1109/TVCG.2013.212

[23] J. Chuang and D. J. Hsu. Human-centered interactive clustering for data analysis. *Conference on Neural Information Processing Systems (NIPS). Workshop on Human-Propelled Machine Learning*, 2014.

[24] C. Collins, G. Penn, and S. Carpendale. Bubble sets: Revealing set relations with isocontours over existing visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1009–1016, Nov 2009. doi: 10.1109/TVCG.2009.122

[25] R. Cordeiro de Amorim and P. Komisarczuk. *On Initializations for the Minkowski Weighted K-Means*, pp. 45–55. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. doi: 10.1007/978-3-642-34156-4_6

[26] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, 2001. doi: 10.1023/A:1007612920971

[27] C. Ding and X. He. K-means clustering via principal component analysis. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pp. 29–. ACM, New York, NY, USA, 2004. doi: 10.1145/1015330.1015408

[28] C. Ding and T. Li. Adaptive dimension reduction using discriminant analysis and k-means clustering. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pp. 521–528. ACM, New York, NY, USA, 2007. doi: 10.1145/1273496.1273562

[29] D. L. Donoho. High-dimensional data analysis: The curses and blessings of dimensionality. In *AMS Conference on Math Challenges of the 21st Century*, 2000.

[30] J. C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57, 1973. doi: 10.1080/01969727308546046

[31] T. Dwyer, Y. Koren, and K. Marriott. Ipsep-cola: An incremental procedure for separation constraint layout of graphs. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):821–828, Sept 2006. doi: 10.1109/TVCG.2006.156

[32] C. F. Eick, N. Zeidat, and Z. Zhao. Supervised clustering - algorithms and benefits. In *16th IEEE International Conference on Tools with Artificial Intelligence*, pp. 774–776, Nov 2004. doi: 10.1109/ICTAI.2004.111

[33] A. Endert, S. Fox, D. Maiti, S. Leman, and C. North. The semantics of clustering: Analysis of user-generated spatializations of text documents. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pp. 555–562. ACM, New York, NY, USA, 2012. doi: 10.1145/2254556.2254660

[34] A. Endert, C. Han, D. Maiti, L. House, S. Leman, and C. North. Observation-level interaction with statistical models for visual analytics. In *2011 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 121–130, Oct 2011. doi: 10.1109/VAST.2011.6102449

[35] A. Endert, M. S. Hossain, N. Ramakrishnan, C. North, P. Fiaux, and C. Andrews. The human is the loop: new directions for visual analytics. *Journal of Intelligent Information Systems*, 43(3):411–435, 2014. doi: 10.1007/s10844-014-0304-9

[36] V. Estivill-Castro. Why so many clustering algorithms: A position paper. *SIGKDD Explor. Newsl.*, 4(1):65–75, June 2002. doi: 10.1145/568574.568575

[37] U. M. Fayyad, A. Wierse, and G. G. Grinstein. *Information visualization in data mining and knowledge discovery*. Morgan Kaufmann, 2002.

[38] I. K. Fodor. *A Survey of Dimension Reduction Techniques*. May 2002. doi: 10.2172/15002155

[39] S. L. France and J. D. Carroll. Two-way multidimensional scaling: A review. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 41(5):644–661, Sept 2011. doi: 10.1109/TSMCC.2010.2078502

[40] J. H. Friedman and J. W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, C-23(9):881–890, Sept 1974. doi: 10.1109/T-C.1974.224051

[41] E. R. Gansner, Y. Hu, and S. Kobourov. Gmap: Visualizing graphs and clusters as maps. In *2010 IEEE Pacific Visualization Symposium (PacificVis)*, pp. 201–208, March 2010. doi: 10.1109/PACIFICVIS.2010.5429590

[42] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, Mar. 2003.

[43] H. H. Harman. Modern factor analysis. 1960.

[44] J. Heer and D. Boyd. Vizster: visualizing online social networks. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pp. 32–39, Oct 2005. doi: 10.1109/INFVIS.2005.1532126

[45] C. Heine and G. Scheuermann. Manual clustering refinement using interaction with blobs. In *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization*, EUROVIS'07, pp. 59–66. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007. doi: 10.2312/VisSym/EuroVis07/059-066

[46] L. House, S. Leman, and C. Han. Bayesian visual analytics: Bava. *Statistical Analysis and Data Mining*, 8(1):1–13, 2015. doi: 10.1002/sam.11253

[47] X. Hu, L. Bradel, D. Maiti, L. House, C. North, and S. Leman. Semantics of directly manipulating spatializations. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2052–2059, Dec 2013. doi: 10.1109/TVCG.2013.188

[48] J. Z. Huang, M. K. Ng, H. Rong, and Z. Li. Automated variable weighting

in k-means type clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):657–668, May 2005. doi: 10.1109/TPAMI.2005.95

[49] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent component analysis*, vol. 46. John Wiley & Sons, 2004.

[50] S. Ingram, T. Munzner, and M. Olano. Glimmer: Multilevel mds on the gpu. *IEEE Transactions on Visualization and Computer Graphics*, 15(2):249–261, March 2009. doi: 10.1109/TVCG.2008.85

[51] R. Jianu, A. Rusu, Y. Hu, and D. Taggart. How to display group information on node-link diagrams: An evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 20(11):1530–1541, Nov 2014. doi: 10.1109/TVCG.2014.2315995

[52] P. Joia, F. Petronetto, and L. G. Nonato. Uncovering representative groups in multidimensional projections. In *Proceedings of the 2015 Eurographics Conference on Visualization*, EuroVis '15, pp. 281–290. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2015. doi: 10.1111/cgf.12640

[53] E. Kandogan. Star coordinate: A multi-dimensional visualization technique with uniform treatment of dimensions. In *Proceedings of the IEEE Information Visualization Symposium*, vol. 650, p. 22.

[54] E. Kandogan. Just-in-time annotation of clusters, outliers, and trends in point-based data visualizations. In *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 73–82, Oct 2012. doi: 10.1109/VAST.2012.6400487

[55] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, Sep 1990. doi: 10.1109/5.58325

[56] H.-P. Kriegel, P. Kröger, and A. Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Trans. Knowl. Discov. Data*, 3(1):1:1–1:58, Mar. 2009. doi: 10.1145/1497577.1497578

[57] H. Lee, J. Kihm, J. Choo, J. Stasko, and H. Park. ivisclustering: An interactive visual document clustering via topic modeling. *Computer Graphics Forum*, 31(3pt3):1155–1164, 2012. doi: 10.1111/j.1467-8659.2012.03108.x

[58] J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2007.

[59] S. C. Leman, L. House, D. Maiti, A. Endert, and C. North. Visual to parametric interaction (v2pi). *PloS one*, 8(3), 2013.

[60] M. Levandowsky and D. Winter. Distance between sets. *Nature*, 234(5323):34–35, 1971.

[61] S. Liu, D. Maljovec, B. Wang, P. T. Bremer, and V. Pascucci. Visualizing high-dimensional data: Advances in the past decade. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1249–1268, March 2017. doi: 10.1109/TVCG.2016.2640960

[62] S. Liu, B. Wang, P.-T. Bremer, and V. Pascucci. Distortion-guided structure-driven interactive exploration of high-dimensional data. *Computer Graphics Forum*, 33(3):101–110, 2014. doi: 10.1111/cgf.12366

[63] S. Liu, M. X. Zhou, S. Pan, W. Qian, W. Cai, and X. Lian. Interactive, topic-based visual text summarization and analysis. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pp. 543–552. ACM, New York, NY, USA, 2009. doi: 10.1145/1645953.1646023

[64] S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, Mar 1982. doi: 10.1109/TIT.1982.1056489

[65] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *J. Mach. Learn. Res.*, 9:2579–2605, Sept. 2008.

[66] G. M. H. Mamani, F. M. Fatore, L. G. Nonato, and F. V. Paulovich. User-driven feature space transformation. *Computer Graphics Forum*, 32(3pt3):291–299, 2013. doi: 10.1111/cgf.12116

[67] A. Mayorga and M. Gleicher. Splatterplots: Overcoming overdraw in scatter plots. *IEEE Transactions on Visualization and Computer Graphics*, 19(9):1526–1538, Sept 2013. doi: 10.1109/TVCG.2013.65

[68] K. Misue, P. Eades, W. Lai, and K. Sugiyama. Layout adjustment and the mental map. *Journal of Visual Languages & Computing*, 6(2):183 – 210, 1995. doi: 10.1006/jvlc.1995.1010

[69] T. Munzner. *Visualization Analysis and Design*. CRC Press, 2014.

[70] E. J. Nam, Y. Han, K. Mueller, A. Zelenyuk, and D. Imre. Clustersculptor: A visual analytics tool for high-dimensional data. In *2007 IEEE Symposium on Visual Analytics Science and Technology*, pp. 75–82, Oct 2007. doi: 10.1109/VAST.2007.4388999

[71] A. Y. Ng, M. I. Jordan, Y. Weiss, et al. On spectral clustering: Analysis and an algorithm. In *NIPS*, vol. 14, pp. 849–856, 2001.

[72] D. Niu, J. G. Dy, and M. I. Jordan. Dimensionality reduction for spectral clustering. In *Proceedings of the 14th International Conference Artificial Intelligence and Statistics*, AISTATS '11, pp. 552–560. ACM, New York, NY, USA, 2011.

[73] F. Paulovich, D. Eler, J. Poco, C. Botha, R. Minghim, and L. Nonato. Piecewise laplacian-based projection for interactive data exploration and organization. *Computer Graphics Forum*, 30(3):1091–1100, 2011. doi: 10.1111/j.1467-8659.2011.01958.x

[74] K. Pearson. Principal components analysis. *The London, Edinburgh and Dublin Philosophical Magazine and Journal*, 6(2):566, 1901.

[75] D. Pelleg, A. W. Moore, et al. X-means: Extending k-means with efficient estimation of the number of clusters. In *ICML*, vol. 1, pp. 727–734, 2000.

[76] P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. *Proceedings of International Conference on Intelligence Analysis*, pp. 2–4, 2005.

[77] B. Rieck and H. Leitte. Persistent homology for the evaluation of dimensionality reduction schemes. *Computer Graphics Forum*, 34(3):431–440, 2015. doi: 10.1111/cgf.12655

[78] D. Ro and H. Pe. *Pattern classification and scene analysis*. John Wiley & Sons, New York, USA, 1973.

[79] B. Saket, P. Simonetto, S. Kobourov, and K. Brner. Node, node-link, and node-link-group diagrams: An evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2231–2240, Dec 2014. doi: 10.1109/TVCG.2014.2346422

[80] J. Z. Self, R. K. Vinayagam, J. T. Fry, and C. North. Bridging the gap between user intention and model parameters for human-in-the-loop data analytics. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, HILDA '16, pp. 3:1–3:6. ACM, New York, NY, USA, 2016. doi: 10.1145/2939502.2939505

[81] P.-N. Tan, M. Steinbach, and V. Kumar. Data mining cluster analysis: basic concepts and algorithms. In *Introduction to data mining*, chap. 8. Pearson Education India, 2013.

[82] J. B. Tenenbaum, V. d. Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. doi: 10.1126/science.290.5500.2319

[83] R. L. Thorndike. Who belongs in the family? *Psychometrika*, 18(4):267–276, 1953. doi: 10.1007/BF02289263

[84] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[85] W. S. Torgerson. Theory and methods of scaling. 1958.

[86] C. Turkay, P. Filzmoser, and H. Hauser. Brushing dimensions - a dual visual analysis model for high-dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2591–2599, Dec 2011. doi: 10.1109/TVCG.2011.178

[87] F. Valafar. Pattern recognition techniques in microarray data analysis. *Annals of the New York Academy of Sciences*, 980(1):41–64, 2002. doi: 10.1111/j.1749-6632.2002.tb04888.x

[88] T. von Landesberger, S. Fiebig, S. Bremm, A. Kuijper, and D. W. Fellner. *Interaction Taxonomy for Tracking of User Actions in Visual Analytics Applications*, pp. 653–670. Springer New York, New York, NY, 2014. doi: 10.1007/978-1-4614-7485-2_26

[89] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl, et al. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, vol. 1, pp. 577–584, 2001.

[90] J. Wenskovitch and C. North. Observation-level interaction with clustering and dimension reduction algorithms. In *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics*, HILDA'17, pp. 14:1–14:6. ACM, New York, NY, USA, 2017. doi: 10.1145/3077257.3077259

[91] A. Wismüller, M. Verleysen, M. Aupetit, and J. A. Lee. Recent advances in nonlinear dimensionality reduction, manifold and topological learning. In *ESANN*, 2010.

[92] R. Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, May 2005. doi: 10.1109/TNN.2005.845141

[93] J. S. Yi, Y. a. Kang, and J. Stasko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, Nov 2007. doi: 10.1109/TVCG.2007.70515

[94] H. Zha, X. He, C. Ding, M. Gu, and H. D. Simon. Spectral relaxation for k-means clustering. In *Advances in Neural Information Processing Systems*, pp. 1057–1064, 2002.