Optimizing Star-coordinate Visualization Models for Effective Interactive Cluster Exploration on Big Data

Keke Chen
Data Intensive Analysis and Computing (DIAC) Lab
Ohio Center of Excellence in Knowledge-enabled Computing (Kno.e.sis)
Department of Computer Science and Engineering,
Wright State University, Dayton OH 45435, USA
keke.chen@wright.edu, (+1)937-775-4642

Abstract

Interactive visual cluster analysis is the most intuitive way for finding clustering patterns, validating algorithmic clustering results, understanding data clusters with domain knowledge, and refining cluster definitions. The most challenging step is visualizing multidimensional data and allowing a user to interactively explore the data to identify clustering structures. In this paper, we systematically study the star-coordinate based visualization models and propose the optimized design that presents the best visualization results and supports the most efficient interaction methods. We explain the intuition behind the models and their link with random projection, and then optimize the visual design in terms of the efficiency of visual presentation and interactive operations. We also discuss the randomized visualization generation method, which can be used to generate batches of meaningful visualization results in parallel for big data. Finally, we present the experimental evaluation of the optimal design of models. This study is critical to generating effective visualization and minimizing the computational cost for visualizing data clusters for big data in the cloud.

keyword Interactive Multidimensional Data Visualization, Visual Cluster Analysis, Star-coordinate Models, Big Data

1 Introduction

With widely deployed Internet applications, sensor networks, and data collection tools, the size of dataset is growing fast. By incorporating users into the analysis loop, interactive visual data analysis has become the most effective exploratory data analysis method for discovering useful information and patterns from large datasets [23]. In particular, interactive cluster visualization techniques have shown unique advantages in visually validating algorithmic clustering results, understanding clusters with domain knowledge, and interactively refining cluster definitions for multidimensional data [18, 5, 19, 22].

Compared to other multidimensional visualization methods, such as scatter-plot matrix [3] and parallel coordinates [18], star-coordinate models [19, 5] are probably the most scalable technique for visualizing large datasets. Scalability here refers to both visual representation and data processing.

1. In spite of different underlying mapping models for star coordinates, such as the Kandogan model [19] and the VISTA model [5], the star-coordinates visual designs share a unique feature. It extends the traditional two or three dimensional coordinate systems to k-dimensional coordinate systems as Figure 1 shows. Apparently, this visual design can present more dimensions than the scatter-plot matrix and parallel coordinates. In fact, scatter-plot matrix and parallel coordinates are often used to explore less than ten dimensions while star coordinates can handle tens of dimensions [6]. In particular, star coordinates can present better cluster visualization formed by multiple dimensions. In contrast, each scatter-plot can only observe two dimensions; the clusters are often visualized with heavy overlapping in parallel coordinates.

2. The star-coordinate visualization can scale up to many points, with the help of density-based representation. Because of the density preserving properties of the underlying mapping models, points can be aggregated and we can only show the density information. This is important when the number of records becomes very large, e.g., tens of millions to billions of records.

3. Most importantly, star-coordinate based cluster visualization methods avoid calculating pairwise distances between the records - rather, it utilizes the property of the underlying mapping model to partially preserve the distance relationship. This is very meaningful to processing big data, where the cost of computing pairwise distances becomes unacceptable.

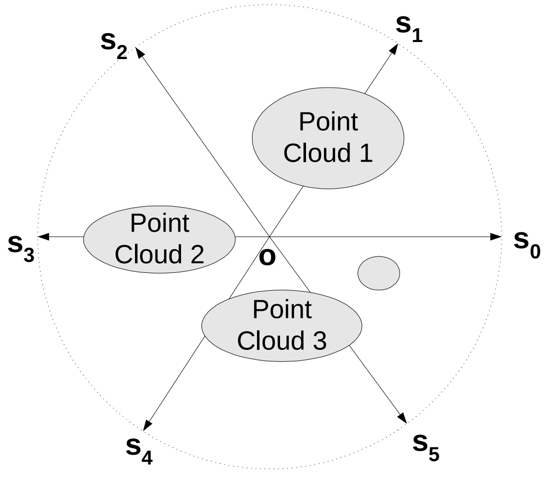

Figure 1: Star Coordinate system with k=6

We describe two important design principles for star-coordinate based cluster exploration. (1) The visualization will not perfectly distinguish all data clusters in ONE frame (if we call a visualization result as a frame)), because the visual distortion is unavoidable when data is projected data from multidimensional space to two-dimensional display. Typically, data clusters are preserved, but they may overlap each other in the visualization. (2) It depends on visual parameter tuning to identify the overlapping in multiple frames. Thus, effective visual and interactive design is required for users to efficient identify the visual overlapping. We believe that effective visual and interactive design is fundamentally related to the underlying mathematical models.

Processing big data[1] puts different requirements on the visual and interactive design. If the system is working on small datasets, for instance, with tens of thousands of records, as shown with current prototypes [5], users' interactions to adjust the visualization can result in almost instant visualization updates. Because of the small size of data, a single workstation can guarantee low response time. However, in the emerging era of big data, the size of data can grow to terabytes, petabytes, or even larger, which cannot be handled by a single workstation. Massive parallel processing infrastructures such as Hadoop/MapReduce [25] have to be used to process data, which are often stored remotely in the cloud [2]. As a result, new algorithms/systems have to generate visual frames with a balance on high-throughput and low-latency. For instance, a batch-interaction

[1]Traditionally, large datasets mean those cannot be easily fit into memory for processing. Thus, database techniques such as indexing are needed. In contrast, big data now refer to the datasets that are so big that existing database techniques feel awkward to handle them. Because of the huge size, big data is often stored and processed in massive parallel processing infrastructure on top of public clouds.

model proposed by our recent work [7] can be applied to address this balance, where batches of visual frames are generated in the cloud and returned to the user for local exploration (Figure 2). This strategy effectively mitigates the conflict between the desired interactivity and the latency caused by remotely processing big data.

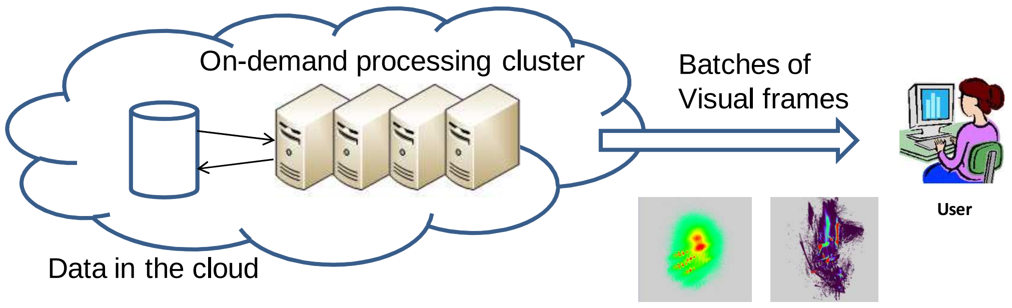

Figure 2: Big datasets are processed in the cloud. Batches of frames are returned to the user, where generating effective useful visual frames is critical to efficient visual exploration.

As visual frames are generated remotely, the system cannot afford to generate too many low-quality frames, which previously can be simply ignored or easily corrected by user's interactions in a local system. Thus, optimizing the visualization models to generate quality visualizations and minimize unnecessary interactions is critical to effective visual exploration of remote big data.

In this paper, we study the mathematical underpinnings of the star-coordinate systems and empirically evaluate the visual performance of different model settings to find the optimal design of star-coordinate models. With the optimal modeling setting, the user now can find more details or clues of clustering structure with higher-quality cluster visualization and fewer interactions.

This paper consists of two components. The first part focuses on formal analysis of the typical star-coordinate models: the Kandogan model [19] and the VISTA model [5], and unifies them to a single representation. Based on the unified model, we describe the major interactive methods that help find visual cluster overlapping. By analyzing the statistical properties of randomly generated visualization results, we show the impact of the different parameter settings of the unified model on the effectiveness of visual exploration. This property also helps the design of the batch-frame processing algorithm RandGen for processing big data in the cloud.

The second part is focused on the experimental study of parameter settings. We design a set of statistical measures to quantify the effectiveness of a dynamically changing cluster visualization, and an algorithm to collect the data and calculate these measures. Concretely, we use `Coverage` to represent the effectiveness of use of visualization space, `Purity` to represent whether visualization results in cluster overlapping, and `Out-of-area Rate` to describe the need for additional adjustment operations. A grid-based data structure is developed to collect the information from dynamically changing the visualization, which is then used to calculate the measures. Experimental results support our analysis on the optimal parameter setting.

The rest of the paper is organized as follows. In Section 2, we briefly review the related visualization techniques for exploratory visual cluster analysis. In section 3, we study the features of star-coordinate based models in the unified model and also develop the intuition for parameter settings. In Section 4, we propose a set of statistical measures based on the micro-structure of visualization for experimental evaluation and design a procedure for collecting these measures. Finally, we present the experiments on parameter settings, based on the proposed measures and the data collecting procedure.

2 Related Work

Interactive multidimensional data visualization is commonly recognized as a useful method for understanding the sophistication in datasets. As an important subarea of data cluster analysis, interactive cluster visualization plays an important role in visually validating and refining the clusters in multidimensional datasets [21, 5].

Interactive visual exploration is a perfect tool for cluster analysis for several reasons. (1) Clustering is fundamentally an iterative exploration process. There is no one algorithm, a similarity measure, or parameter setting that can fit all applications. Users have to try different settings to find the result that matches the application requirements best. (2) Users often have domain knowledge that can guide cluster analysis, and thus need a tool to conveniently incorporate the domain knowledge. (3) Clustering structures might be very irregular (e.g., non-spherical or some kind of manifolds), which need users to understand, validate, and refine, where visualization can be convenient.

Many efforts have been made to visualize clusters. The early research on plot-based data visualization is Grand Tour and Projection Pursuit [9]. Since there are numerous projections from a multidimensional data space to a 2D space, the purpose of the Grand Tour and the Project Pursuit is to guide the user to find "informative projections". It automatically interpolates several pre-selected projections to generate a continuous change of visualization. Yang [26] extends the

Grand Tour technique to show projections in an animation.

The VISTA system [5] has shown that interpolation of visualizations might not be necessary that user intuition and interaction can be nicely integrated into cluster exploration to effectively find informative cluster visualization. Star-coordinate based models [19, 5, 22, 17] in general, including the VISTA model, are particularly good for cluster visualization for the simplicity of the model and the convenience of interaction design. In particular, linear mapping models are often used in these systems, which satisfies the performance requirement of visualizing very large datasets.

Other density-based visualization systems include HD-Eye [16] and OPTICS [1]. OPTICS works well in finding the 1D sketch of arbitrarily shaped clusters. However, 1D visualization provides less information and less flexibility than 2D visualization such as the star-coordinates based models. Most other visualization techniques, such as Scatterplot matrices, coplots, prosection [8] and FastMap based visualization [14] are only good for generating static cluster visualizations for low-dimensional data.

3 Star-coordinate Models and Unification

Star-coordinate based visualization models have been successfully used in cluster analysis and validation, as in the Kandogan system [19] and in the VISTA system [5]. Although the definitions of different models look quite different, they are inherently related because of the use of the same star-coordinate system. In this section, we first describe these two specific models and then unify them into one general model. The unified model allows us to conveniently study the features shared by similar star-coordinate models and develop concise analysis. With the unified model, we analyze the principles of the interactive exploration on star-coordinate visualization and discuss the relationship between the parameter setting and the effectiveness of interactive visualization.

To begin with, we define the notations used in this paper. A dataset is defined as a table with k columns (dimensions) and n rows (records, points, or vectors). A row is represented as a point in the visualization. We use bold lower cases to represent vectors and normal lower cases to represent scalars.

3.1 Unifying Star-Coordinate Models

A star-coordinate based visualization system arranges multiple coordinates in a "star" shape over the display area. As Figure 1 shows, 'o' is the visual center and $\mathbf{s}_i$ are coordinates. Each coordinate corresponds to one particular dimension and a multidimensional data record is mapped to one point on the star coordinate system through certain mathematical mapping model. If the mapping model preserves density, clusters in the original space will be mapped to point clouds on star coordinates. Figure 5 is an implementation of the star-coordinate system.

In the following, first, we give the definition of the two popular mapping models, and then present the unified model. With this unified model, it is easier to conduct the analysis in the class of linear star-coordinate models and understand the features shared by these models.

The VISTA model. The VISTA model has two steps: (1) Normalization. A multidimensional point is normalized to eliminate the bias brought by different value ranges for different columns. (2) The normalized data is transformed using the α-mapping function.

Normalization is used to unify value ranges of columns. There are a few common methods, such as the max-min normalization and the standardization method [11]. VISTA uses the max-min normalization. Let $[min_i, max_i] i = 1 \ldots k$ be the bounds of the column (or dimension) i, $min_i = \min\{$ values in column $i\}$ and $max_i = \max\{$ values in column $i\}$. VISTA model requires the values are normalized to the range [-1,1]. Let v be the original value and v' be the normalized.

$$v' = \frac{2(v - min_i)}{max_i - min_i} - 1 \tag{1}$$

α-mapping is a parameterized linear mapping designed for star coordinates. Let 2D point $Q(x, y)$ represent the map of a k-dimensional (k-D) normalized data point $P(x_1, \ldots, x_k)$ in visualization. Let $\mathbf{s}_i$ be the unit vector of the coordinate i in the star coordinate system, i.e., $\mathbf{s}_i = (cos(\theta_i), sin(\theta_i))$ and θ_i be the angle of the coordinate i. The contribution of dimension i of the point P to the visual position Q is defined as $\alpha_i x_i \mathbf{s}_i$, where the parameter α_i is used to weight the contribution. Suppose that $\mathbf{o} = (x_0, y_0)$ is the center of the display area and c is a visual scaling parameter. α-mapping is defined as follows.

$$A(x_1, \ldots, x_k, \alpha_1, \ldots, \alpha_k, \theta_1, \ldots, \theta_k, c, \mathbf{o}) = c \sum_{i=1}^{k} \alpha_i x_i' \mathbf{s}_i - \mathbf{o} \tag{2}$$

or, $Q(x, y)$ is represented as

$$Q(x, y) = (c \sum_{i=1}^{k} \alpha_i x_i' \cos(\theta_i) - x_0, c \sum_{i=1}^{k} \alpha_i x_i' \sin(\theta_i) - y_0) \tag{3}$$

where x_i' is the normalized dimensional value and α_i is in the range [-1,1].

The Kandogan Model. With the same definitions of $Q(x, y)$, x_i, min_i, and max_i in the

VISTA model. The mapping function used in paper [19] is defined as follows.

$$Q(x,y) = (\sum_{i=1}^{k}(x_i - min_i)u_{xi}, \sum_{i=1}^{k}(x_i - min_i)u_{yi}) \tag{4}$$

where,

$$\mathbf{u}_i = (u_{xi}, u_{yi}) = \frac{\mathbf{d}_i}{max_i - min_i}$$

max_i and min_i are the actual value bounds as defined by max-min normalization. $\mathbf{d}_i$ is the scalable axis along the i-th dimension, and both of its length and direction can be changed interactively in the visualization system.

The Unified Model. For easier investigation of the models, we unify the above two models. Let's examine the meaning of the Kandogan model and convert it to the VISTA representation. $\mathbf{d}_i$ in the Kandogan model is related to $\mathbf{s}_i$ in the VISTA model. Since $\mathbf{d}_i$ is scalable, let's represent it as $\mathbf{d}_i = \alpha'_i\mathbf{s}_i$, where $\alpha'_i \in [0,\infty)$. Plugging in the definition of u_{xi} and u_{yi}, to Eq. 4, we get

$$Q(x,y) = (\sum_{i=1}^{k}(x_i - min_i)(\frac{\alpha'_i\cos(\theta_i)}{max_i - min_i}), \sum_{i=1}^{k}(x_i - min_i)(\frac{\alpha'_i\sin(\theta_i)}{max_i - min_i})) \tag{5}$$

Let $x'_i = \frac{x_i - min_i}{max_i - min_i}$, which is the form of min-max normalization. We get the final form similar to α-mapping in the VISTA model.

$$Q(x,y) = (\sum_{i=1}^{k} x'_i\alpha'_i\cos(\theta_i), \sum_{i=1}^{k} x'_i\alpha'_i\sin(\theta_i)) \tag{6}$$

There are two differences from the VISTA model. (1) The definition of x'_i actually normalizes the original value to the range [0,1]. (2) The α' parameters have a range of $[0,\infty)$. It is equivalent to use $c \in [0,\infty)$ and $\alpha' \in [0,1]$ to replace $\alpha' \in [0,\infty)$, which results in the exact form of the VISTA model. With the above understanding, we define the unified model as follows. It includes two steps:

1. *Normalization* scales value v to a value v' in the range $[\eta_1,\eta_2]$. The setting of η_1 and η_2 is different from model to model.

$$v = (\eta_2 - \eta_1)\frac{v - min_i}{max_i - min_i} + \eta_1. \tag{7}$$

2. The second step uses *α-mapping* (Eq. 2) with α range in $[\zeta_1,\zeta_2]$. The setting of ζ_1 and ζ_2 is different from model to model.

To instantiate the unified model, the VISTA model has $\eta_1 = -1, \eta_2 = 1, \zeta_1 = -1$, and $\zeta_2 = 1$, while the Kandogan model has $\eta_1 = 0, \eta_2 = 1, \zeta_1 = 0$, and $\zeta_2 = 1$.

3.2 Properties of the Unified Model

A good understanding of the unified model will help us better understand the parameter settings and their effect on visualization and interaction. Below we will give some intuition about the unified model, which also applies to the VISTA model and the Kandogan model. It will also explain why these models are good for exploring clustering structures.

First, we look at the implication of a single static visualization to data clustering. With all parameters fixed, we can represent the mapping with a standard matrix form in terms of the normalized input data $\mathbf{x}' = (x'_1, \ldots, x'_k)$,

$$G(\mathbf{x}') = A\mathbf{x}' + \mathbf{o}, \tag{8}$$

where A is a constant matrix and

$$A = c(\mathbf{s}_1, \ldots, \mathbf{s}_k) \begin{pmatrix} \alpha_1 & & \\ & \ddots & \\ & & \alpha_k \end{pmatrix}. \tag{9}$$

Because the max-min normalization can be represented as an affine transformation as well, the G() transformation is also an affine mapping in terms of the original data vectors. We will use this generalized form to analyze the properties. Based on this setting, the unified model is essentially a simple linear model with dimensional adjustable parameters α_i. The rationale behind the model is

Proposition 3.1 *If Euclidean distance is used as the similarity measure, an affine mapping does not break clusters but may cause cluster overlapping.*

Proof. Let's model arbitrarily shaped clusters with a Gaussian mixture [12]. Let μ be the density center and Σ be the covariance matrix of the Gaussian cluster. A cluster C_i is an ellipsoid, which can be represented with

$$\mathcal{N}_i(\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{k/2}|\Sigma_i|^{1/2}} \exp\{-(\mathbf{x} - \mu_\mathbf{i})'\Sigma^{-1}(\mathbf{x} - \mu_\mathbf{i})/2\}$$

Geometrically, μ describes the position of the cluster and Σ describes the spread of the dense area. After an affine transformation, say $G(\mathbf{x}) = A\mathbf{x} + \mathbf{b}$, the center of the cluster is moved to $A\mu_i + \mathbf{b}$ and the covariance matrix (corresponding to the shape of dense area) is changed to $A\Sigma_i A^T$ as illustrated by Figure 3. And the dense area is modeled with $\mathcal{N}_i(A\mu_i + \mathbf{b}, A\Sigma_i A^T)$. Therefore, affine mapping does not break the dense area, i.e., the cluster. However, due to the changed shapes of the

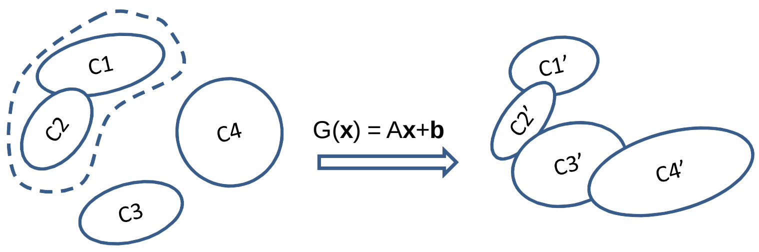

Figure 3: Use a Gaussian mixture to describe the clusters in the dataset.

clusters, $A\Sigma_i A^T$, some clusters may overlap each other. As the unified model is an affine model, this proposition also applies to the unified model.$\square$

Since there is no "broken cluster" in the visualization, any visual gap between the point clouds reflects the real density gaps between the clusters in the original high-dimensional space. The only challenge is to distinguish the distance distortion and cluster overlapping introduced by the mapping. Uniquely different from other models, by tuning α_i values, we can scrutinize the multidimensional dataset visually from different perspectives, which gives dynamic visual clues for distinguishing the visual overlapping[2].

Non-Euclidean Distances. Because human visual perception is very sensitive to Euclidean distance, in particular the clusters (or dense areas) formed by Euclidean distance, other metric spaces will need to be converted to Euclidean spaces. This line of research is known as Euclidean embedding [4, 24]. We have been working on scalable embedding algorithms, such as the MapReduce implementation of Landmark MDS (LMDS) [10] and FastMap [14], which can serve as the pre-processing step. However, this topic is out of the scope of this paper. Thus, we will not explore in more details.

4 Optimal Design of Star-Coordinate Visualization

As we have discussed, the unified model is clearly an affine mapping [15] from multidimensional space ($> 3D$) to two-dimensional space. If Euclidean distance is used dissimilarity measure for clustering, which is also the most common case, these dense areas are clusters [13]. The major problem of the linear models is partial cluster preserving, which means the dense areas are preserved, but they may overlap each other in visualization. In the following, we discuss how to design

[2]A well-known problem is that the linear model cannot visually separate some manifold structures such as nested spherical surfaces, which can be addressed by using spectral clustering [20] as the preprocessing step.

interactive operations to help distinguish the visual cluster overlapping.

The discussed model has a rich set of tunable parameters, i.e., the normalization ranges η_1 and η_2, α_i, α's range, θ_i, and c, which gives many possibilities to interactively generate visualization. However, having too many possible parameter tuning methods may also overload the user visual perception and interaction. We address this problem from three aspects. (1) Among all of the possible tuning methods, what are the minimal set of the operations that are necessary to achieve the target of finding visually separated clusters? (2) Can we use heuristic rules to minimize the number of interactions? (3) Will different parameter range settings affect the effectiveness of visualization? We will discuss these problems in detail.

4.1 The Minimal Set of Necessary Interactive Operations

A typical visual design enables the convenient manipulation of the three sets of parameters, α_i, θ_i and c. Figure 4 shows the VISTA visual design. The α-widgets (the blue dots) are used to interactively adjust the value of α values. Clicking on these α-widgets, the user can change the α values. α-gauges (the colored rays from the display center) represent the scale of α value. α-widgets can also be dragged around the center to change the θ parameters. These visual designs are naturally combined with the star coordinates. We categorize these operations into three basic categories: *α-adjustment* to adjust α_i values, *zooming* to scale c, and *rotating* to change the direction of coordinates.

The Kandogan system has some similar design for these three types of operations. With the increase of dimensionality, the number of parameter combinations will increase exponentially. As a result, the user might feel difficult to use such a system. Our first effort is to identify a reduced set of operations, with which we can still achieve the same goal of using the whole set of operations.

The first method is to eliminate the rotation operation that changes the θ values from the necessary set of operations. We notice that rotating is equivalent to a pair of α-adjustment operations.

Figure 6 illustrates how one rotating operation can be decomposed to a pair of α-adjustment operations. Assume the $\mathbf{s}_i$ and $\mathbf{s}_{i+1}$ are two neighboring coordinates. Let $\alpha_i\mathbf{s}_i$ and $\alpha_i\mathbf{s}_i$ be the current weighted dimensional contribution. If $\mathbf{s}_i$ is rotated by an angle to $\mathbf{s}'_i$, $\mathbf{s}'_i$ can be decomposed to two vectors $u\alpha_i\mathbf{s}_i$ and $v\alpha_i\mathbf{s}_{i+1}$. It is thus equivalent to two α adjustment operations: reducing the α_i to $u\alpha_i$ and increasing α_{i+1} to $\alpha_{i+1} + v\alpha_i x_i/x_{i+1}$, since the original α_{i+1} is unchanged. For the case (Figure 6 right) that the neighboring coordinate $\mathbf{s}_{i+1}$ has an angle greater than 180^o to $\mathbf{s}_i$ along the side of coordinate movement, v turns to negative.

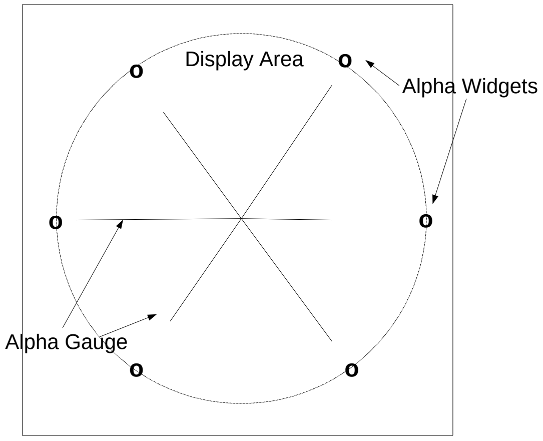

Figure 4: A typical visual design of star-coordinate visualization.

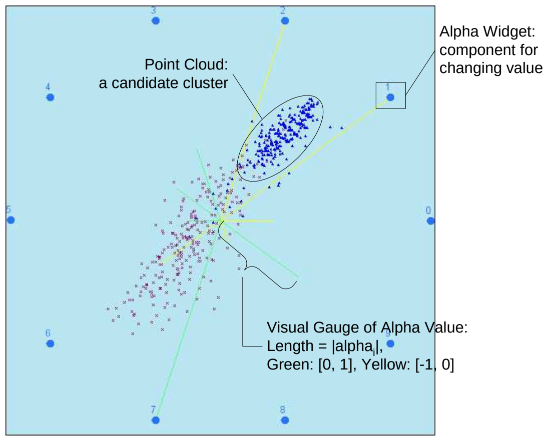

Figure 5: An implementation of the star coordinate system (the VISTA system).

Since zooming does not really change the visual density distribution, but rather changes the detail level of the visualization, the most significant operation of finding the visual overlapping is only one type of operation: α-adjustment.

4.2 Heuristic Rules for Interactive Exploration

There is the practical meaning of α-adjustment. Large α_i values emphasis on the weight of the i-th dimension to the visual clustering results. However, the number of possible combinations of α-adjustments still too large for high dimensionality. We recognize that visual exploration is not a process of simply enumerating all possible parameter settings – more importantly, we want to incorporate user's visual intuition into the exploration process.

We introduce the intuition of exploration and then give some heuristic rules to help significantly improve the efficiency of exploration. Dense areas in the original space are visualized as point clouds in star-coordinates visualization and changing α values causes the moving of point clouds. In interactive visual rendering, some dimensions often show some "significant change" to visualization in a series of continuous α-adjustment, i.e., changing α_i value results in distinct point clouds moving in different directions, or causes the emergence of visible "gaps" between point clouds (Figure 7). Intuitively, these dimensions play important roles in distinguishing the visual overlapping and thus we name them as "the visually dominating dimensions", and also the others as "the fine-tuning dimensions". In practice, the following heuristic rules can be used.

1. Sequentially try rendering each dimension, if the dimension is a visually dominating dimen-

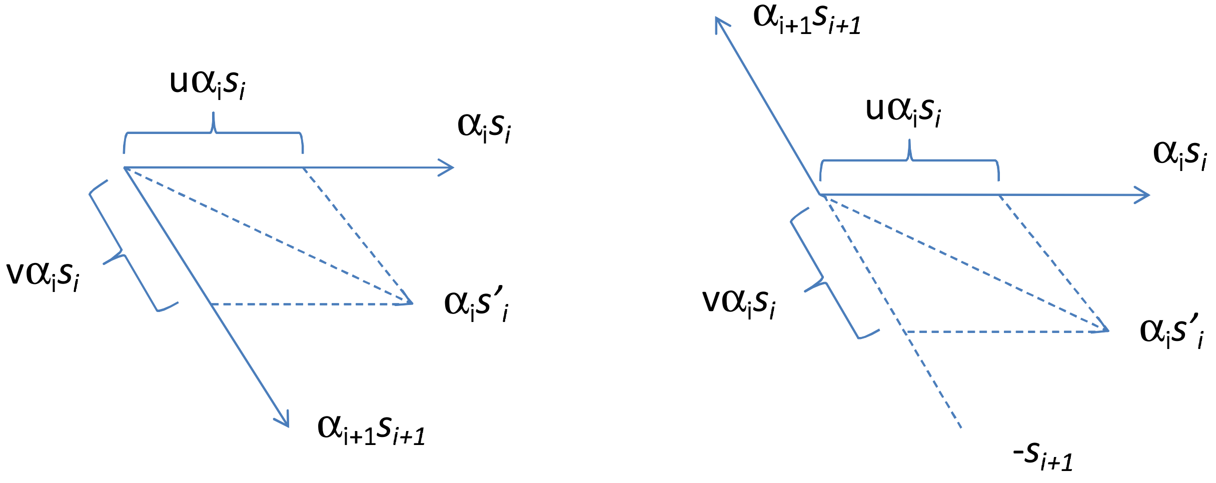

Figure 6: Decomposition of axis rotation

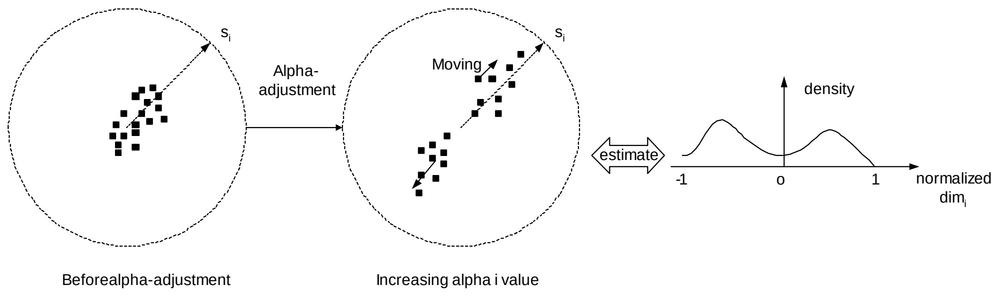

Figure 7: Continuous α-adjustment

sion, adjust its α value to a certain degree so that the point clouds are satisfactorily separated.

2. Use the fine-tuning dimensions to polish the visualization. Adjust their α values finely so that the visualization clearly shows the well-separated point clouds.

These rules have heavily depended on the effectiveness of visualization. For example, we want α-adjustment of dominating dimensions to easily separate the point clouds; we want fine tuning can result in visualization with sufficient details. These problems might be related to the initial setting of the normalization range and the α value ranges. We discuss these parameter settings in the next section.

4.3 Statistical Properties of Randomized Exploration and Its Application to Big Data

In the single workstation mode for medium-size data, the workstation can quickly respond to user's interactive operation and re-generate the visualization by applying the star-coordinate model to the entire dataset or sample data. However, this interaction model is not realistic if the dataset is so big that it has to be hosted remotely, for instance, in the cloud [2]. In this case, we may use a batch-interaction method: the remote server generates a batch of relevant visualization results; the results, which have a much smaller size than the dataset, are sent back to the local user; then, the user interactively explore them. There is a critical problem in this batch-interaction method - how to generate meaningful batches of visualizations.

In this section, we discuss the frame-based random visualization generation algorithm RandGen and show that its statistical properties can help a user effectively identify the visual cluster overlapping. We define *a frame* as a visualization result according to one setting of the model parameters. Since we only need to encode the density information of a frame, we represent a frame with a two-dimensional density map, where each cell represents a coordinate and records the number of points that are mapped to the cell.

The RandGen algorithm is a random perturbation process that generates a collection of related frames. According to the previous discussion, we tune only the α parameters to update the visualization. Starting from the initial α values that are given by the user, RandGen applies the following small stochastic updates to all dimensional weights simultaneously. Let α_i^{ϕ} represent the α parameter for dimension i in frame ϕ, the new parameter $\alpha_i^{\phi+1}$ is defined randomly as follows.

$$\begin{aligned} \delta_i &= t \times B, \\ \alpha_i^{\phi+1} &= \begin{cases} 1 & \text{if } \alpha_i^{\phi} + \delta_i > 1 \\ \alpha_i^{\phi} + \delta_i & \text{if } \alpha_i^{\phi} + \delta_i \in [-1, 1] \\ -1 & \text{if } \alpha_i^{\phi} + \delta_i < -1, \end{cases} \end{aligned} \tag{10}$$

where t is a predefined step length, often set to small values, e.g., $0.01 \sim 0.05$, and B is a coin-tossing random variable - with probability 0.5 it returns 1 or -1. δ_i is generated independently at random for each dimension. $\alpha_i^{\phi+1}$ is also bounded by the range [-1,1] to minimize the out-of-bound points (those mapped out of the display). This process repeats until the α parameters for the desired number of frames are generated. Since the adjustment at each step is small, the change between the neighboring frames is small and smooth. As a result, sequentially visualized these frames will create continuously changing the visualization. The following analysis shows why the RandGen algorithm can help identify visual cluster overlapping.

Identifying Clustering Patterns with RandGen. We formally analyze why this random perturbation process can help us identify the clustering structure. The change of visualization by adjusting α values can be described by the random movement of each visualized point. Let $\mathbf{v}_1$ and $\mathbf{v}_2$ be the images of the original data record $\mathbf{x}$ for the two neighboring frames, respectively. Then, the point movement is represented as

$$\Delta_{\mathbf{u}} = c\sum_{i=1}^{k} \delta_i x_i \mathbf{s}_i.$$

By definition of B, we have $E[\delta_i] = tE[B] = 0$. Since δ_i are independent of each other, we derive the expectation of $\delta_i\delta_j$

$$E[\delta_i\delta_j] = E[\delta_i]E[\delta_j] = 0, for\ i \neq j.$$

Thus, it follows the expectation of point movement is zero: $E[\Delta_{\mathbf{u}}] = 0$. That means the point will randomly move around the initial position. Let the coordinate $\mathbf{s}_i$ be (s_{i1}, s_{i2}). We can derive the variance of the movement $\text{var}(\Delta_{\mathbf{u}}) =$

$$c^2t^2\text{var}(B)\sum_{i=1}^{k} x_i^2 \begin{pmatrix} s_{i1}^2 & s_{i1}s_{i2} \\ s_{i1}s_{i2} & s_{i2}^2 \end{pmatrix} \tag{11}$$

There are a number of observations based on the variance. (1) The larger the step length t, the more actively the point moves; (2) As the values s_{ix} and s_{iy} are shared by all points, the points with larger vector length $\sum_{i=1}^{k} x_i^2$ tends to move more actively.

Since we want to identify the cluster overlapping by observing point movements, it is meaningful to see how the relative point positions change. Let $\mathbf{w}_1$ and $\mathbf{w}_2$ be the images of another original data record $\mathbf{y}$ for the neighboring frames, respectively. With the previous definition of $\mathbf{x}$, the visual squared distance between the pair of points in the initial frame will be

$$\Delta_{\mathbf{w},\mathbf{v}}^{(1)} = ||\mathbf{w}_1 - \mathbf{v}_1||^2 = ||c\sum_{i=1}^{k} \alpha_i(x_i - y_i)\mathbf{s}_i||^2. \tag{12}$$

Then, the change of the squared distance after the perturbation is

$$\begin{aligned}
\Delta_{\mathbf{w},\mathbf{v}} &= 1/c^2(\Delta^{(2)}_{\mathbf{w},\mathbf{v}} - \Delta^{(1)}_{\mathbf{w},\mathbf{v}}) \\
&= (\sum_{i=1} \delta_i(x_i - y_i)s_{i1})^2 + (\sum_{i=1} \delta_i(x_i - y_i)s_{i2})^2 \\
&+ 2(\sum_{i=1} \delta_i(x_i - y_i)s_{i1})(\sum_{i=1} \alpha_i(x_i - y_i)s_{i1}) \\
&+ 2(\sum_{i=1} \delta_i(x_i - y_i)s_{i2})(\sum_{i=1} \alpha_i(x_i - y_i)s_{i2}).
\end{aligned}$$

With the independence between δ_i and δ_j for $i \neq j$, $E(\delta_i) = 0$, $\mathbf{s}^2_{i1} + \mathbf{s}^2_{i2} = 1$, and $E^2[\delta_i] = t^2\text{var}(B) = 0.25t^2$, it follows the expectation of the distance change is

$$E[\Delta_{\mathbf{w},\mathbf{v}}] = \sum_{i=1}^{k} E^2[\delta_i](x_i - y_i)^2 = 0.25t^2 \sum_{i=1}^{k}(x_i - y_i)^2,$$

i.e., the average change of distance is proportion to the original distance between the two points. That means, if points are distant in the original space, we will have higher probability to see them distant in the visual frames; if the points are close in the original space, we will more likely observe them move together in the visual frames. This dynamics of random point movement helps us identify possible cluster overlapping in a series of continuously changing visual frames generated with the RandGen method.

4.4 Parameter Range Setting and the Effect of Interaction

The two settings of parameter range: $[\eta_1, \eta_2]$ for max-min normalization and $[\zeta_1, \zeta_2]$ for α ranges, will have significant impact on the resultant visualization and interactions. We use the VISTA model's setting (normalization range $[-1, 1]$ and α range [-1, 1]), and the Kandogan system's setting (normalization range [0, 1] and α range $[0, 1]$), for example.

Because the goal of exploration is to distinguish the visually overlapped clusters, we want to maximize the utility of each interaction (e.g.,α parameter tuning) towards the goal. The effectiveness of interaction is especially important when we need to generate visualization results for big data in the cloud, as it will save computations in the cloud [2]. There are some heuristics guiding our design of the model. (1) We want the display space is fully utilized and the visualization are located in the center of the display area so that the possible overlapping details are unfolded. (2) We also want to reduce the chance of mapping points out of the display area, because otherwise additional adjustments (interactions) will be needed.

We assume the data points are samples from a joint multidimensional distribution. Let $\mathbf{x}$ represent a random variable of that distribution. Correspondingly, the mapped results have a two-dimensional distribution. Let $\mathbf{y}$ represent a random variable of the two-dimensional distribution. To align the visualization with the center is equivalent to align the two-dimensional distribution to mean 0, which means $E[\mathbf{y}] = 0$. Let's assume tuning the α parameters is independent of the data distribution[3]. Now applying the unified model, we get

$$E[\mathbf{y}] = c \sum_{i=1}^{k} E[\alpha_i] E[x_i] \mathbf{s}_i. \tag{13}$$

Thus, to have $E[\mathbf{y}] = 0$, we need either $E[x_i] = 0$ or $E[\alpha_i] = 0$. Apparently setting the normalization range to $[-1, 1]$ satisfies the requirement $E[x_i] = 0$. While the normalization range $[0, 1]$ results in skewed visualization, which inefficiently uses the display area. $E[\alpha_i] = 0$ indicates that the random changes of visualization are evenly distributed to all directions around the center, which also efficiently use the display space.

Tuning α in the range [-1, 1] also brings more dynamic information. As shown in Figure 7, let's assume the distribution of the target dimension i has two modes with center at $x_{i,1}$ and $x_{i,2}$, $x_{i,1} < x_{i,2}$, respectively. With an adjustment $\Delta\alpha_i$, the movements along the axis i are $x_{i,1}\Delta\alpha_i$ and $x_{i,2}\Delta\alpha_i$, respectively, and the distance between the two modes is $(x_{i,1} - x_{i,2})\Delta\alpha_i$. Thus, increasing $\Delta\alpha_i$ will separate them, and decreasing will contract them. Changing $\Delta\alpha_i$ to $-\Delta\alpha_i$ will map the two modes to the mirror position to their original positions with $\Delta\alpha_i$. Thus, changing α_i continuously in [-1, 1] will create a "rotation" animation, which provides more vivid information to the user.

The above discussion has given the statistical justification and the intuition how the parameter ranges can affect the visualization in terms of the utilization of display area and the effect of interactive operations, both of which are important factors in interactive cluster visualization. We will empirically explore how these settings can affect the efficiency and effectiveness of star-coordinate based cluster exploration. For convenience, we use $([\zeta_1, \zeta_2], [\eta_1, \eta_2])$ to notate the settings of α range and normalization range for a specific model. For example, ([-1, 1], [-1,1]) represents the VISTA model. In the following section, we will present an evaluation approach to empirically studying the effect of the parameter range settings.

[3]It is true for the RandGen algorithm. In interactive exploration, they are actually highly related. But for simplicity of analysis, we just make this assumption.

5 Experiments on the Effect of Parameter Settings

The above discussion on the parameter range setting has given the intuition of the impact of different range settings on visualization results. In this section, we design experiments to evaluate how the range settings affect the visualization. First, we design a number of statistical measures that approximately capture the different aspects of the effectiveness of interaction and visualization. We also design the data structure and the algorithm for collecting these measures. Finally, we run experiments on a number of real datasets to compare the settings based on the measures.

5.1 Design of Empirical Statistical Measures

A good design of the model will allow us to see more details in each visualization result, with less auxiliary adjustments. Based on these observations, we design the following pixel-based measures.

- *Coverage* is the percentage of pixels in the display space covered by data points, which is used to roughly represent the utilization of display space. The higher the Coverage Rate is, the more details are unfolded, helping easier identification of overlapping.
- *Purity* is the percentage of pixels covered by points from the same cluster. Again, this helps visually identify the clusters and cluster overlaps.
- *Out-of-area Rate* is used to evaluate the percentage of visual points mapped out of the effective display area. Higher out-of-area rate implies that more auxiliary operations (e.g., zooming and translating) are needed to adjust the visualization.

These rates are to be collected and estimated from the experiment. We design a data structure and a data collection procedure for this experiment.

5.2 Procedure for Collecting Data and Calculating the Metrics

Micro Structure. To calculate the first three rates, we build up the *micro-structure* of visualization (Figure 8). The entire display area is divided into column-by-row cells, where each cell corresponds to a pixel or (a block of pixels) of the display area. Multiple k-dimension points may be mapped to one cell. Suppose these points from different data clusters (dense areas) are labeled with different cluster ID (CID). We use two indicators for each cell to describe the possible situations. One is *topCID*, which records the cluster label of the k-D point mapped to the top of the cell. The user can see these top points on the visualization but not those under the top points. All cells

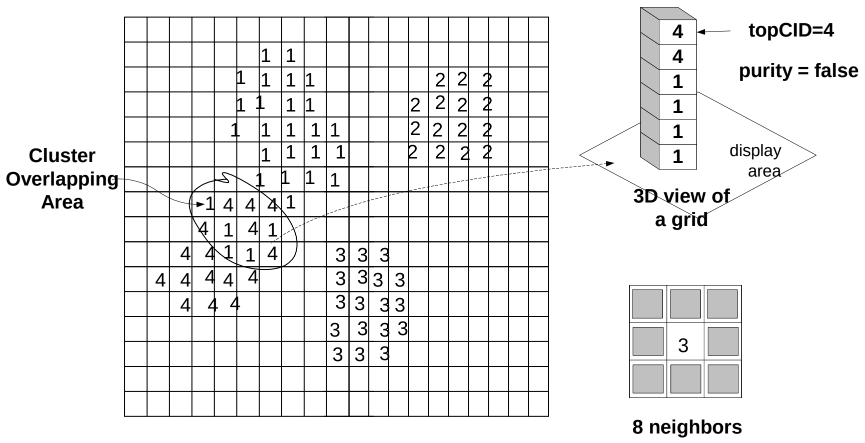

Figure 8: The micro-structure of visualization

covered by points are represented by a positive integer (*topCID*> 0), and empty cells have *topCID* $= 0$. The other indicator is the *Purity* of the cell, which is a boolean variable – *true* represents that all points mapped to this cell have the same cluster label, and *false* otherwise.

With the micro-structure, we can easily calculate the Purity Rate and Out-of-area Rate by counting the cells in one snapshot of visualization during interactive exploration. Let the number of cells with (*purity* = *true*) be p, the total number of cells be C and the cells with *topCID* $= 0$, i.e., empty cells, be C_0.

$$Purity = p/(C - C_0). \tag{14}$$

Let the number of points mapped out of the grid be t, and the total number of points be N.

$$Out\text{-}of\text{-}area\ Rate = t/N. \tag{15}$$

However, *Coverage* cannot be simply evaluated by the number of covered cells. Intuitively, with the same number of covered cells, sparsely distributed cells are easier to tell the details, which is also easier to distinguish visual cluster overlaps. Therefore, we include eight neighbor cells of a covered cell as its contribution to the coverage rate, which takes the sparsity into account. Figure 8 shows the micro-structure and eight neighbors of a cell. The calculation of coverage rate consists of two steps: the first step is labeling the covered cells and their 8-neighbor cells; after labeling, the percentage of labeled cells among all cells is calculated as the *Coverage*.

Apparently, the zooming factor will significantly affect the above rates. Visualization with small zooming factor c will map the data to a small area, resulting in both small *Coverage* and *Purity*; if c is too large, the *Out-of-area rate* may increase. To fairly evaluate these rates crossing different parameter settings, we need to properly adjust the zooming factor so that the visualization is extended enough but without too many points out of the display. This adjustment can be done by the following "automatic zooming" procedure.

Procedure for Collecting the Data.

Automatic zooming (AutoZoom) is designed to make the *Out-of-area Rate* within [0, 0.1%]. It is used to initially set appropriate c; After each interaction changes α parameters, it is used to adjust the visualization if the out-of-area rate is higher than the range. Let R_c be the changing rate of c and R_o be the *Out-of-area Rate*. The following describes the AutoZoom algorithm.

```
1). while (($R_o$ == 0 or $R_o > 0.1\%$) and $R_c > \min_{R_c}$){
2).     while($R_o$ ==0){
3).         $c = c + R_c$;
4).         updating visualization and re-calculating $R_o$; }
5).     $R_c = R_c/2$;
6).     while($R_o > 0.1\%$){
7).         $c = c - R_c$;
8).         updating visualization and re-calculating $R_o$; }
9). }
```

Step 2) to 4) scales up the visualization to make points fill up the whole display area, while Step 5) to 8) scales down to reduce the Out-of-area rate until it falls in the desired range.

We run the RandGen algorithm to generate randomized continuously changing visual frames. Initially, we use the AutoZoom algorithm to appropriately set c. At each step, after α values are updated, the visualization is regenerated. We collect the *Out-of-area rate*, apply AutoZoom again, and then calculate the *Coverage* and the *Purity*. After repeating a number of steps, we average the collected rates.

5.3 Experimental Setup

We will evaluate the three rates for two typical parameter range settings: ([-1,1], [-1,1])) by VISTA model [5] and ([0, 1], [0, 1]) by the Kandogan model [19], using the designed procedure.

Datasets. We use five real datasets for evaluation. The clusters in these datasets are already labeled. These datasets are wisc-breast-cancer, satellite-image, image-segmentation, page-block and shuttle, all from UCI machine learning database[4]. Wisc-breast-cancer is a small dataset with

[4]http://www.ics.uci.edu/~mlearn/ Machine-Learning.html

Dataset	N	k	n_c
Wisc-brest-cancer(WBC)	699	10	2
Page-blocks(PAGE)	5473	10	5
Satellite-image(SAT)	4435	36	5
Image-segmentation(SEG)	2310	19	5
Shuttle.test(SH)	14500	9	7

Table 1: Description of experimental datasets

two unclearly separated clusters. Shuttle is a relatively large dataset, having 3 big clusters in irregular shapes. Other three datasets also have irregular clusters with heavy cluster-overlapping in visualization. We list some detailed descriptions in Table 1, where N is the number of points, k is the number of dimensions, and n_c is the number of labeled clusters.

5.4 Discussion on Experimental Results

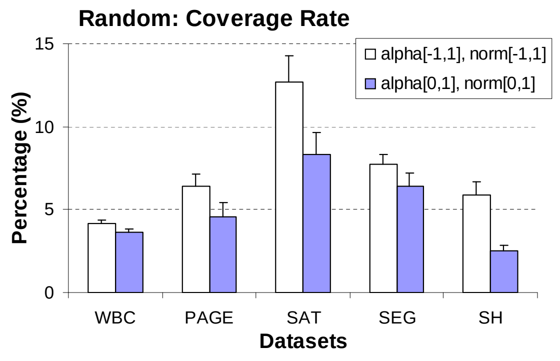

Figure 9: Comparison of coverage rates in random rendering

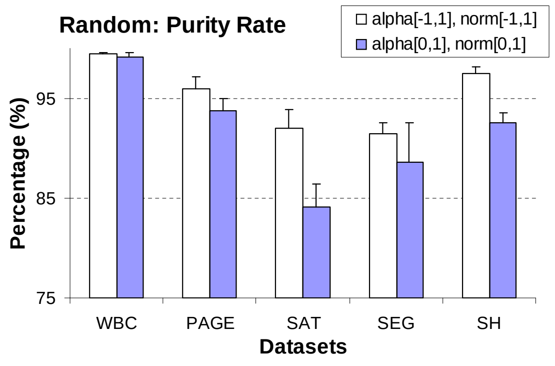

Figure 10: Comparison of purity rates in random rendering

Impact of Parameter Ranges. We run 10 rounds of experiments for each setting of model parameters for the same dataset. Each round of experiments starts a randomly chosen α values and consists of 100 steps of the RandGen algorithm. We use multiple rounds to remove the side-effect of the initial parameter setting. The average measures of the 100 snapshots are used as the estimated rates in one round of experiments. Finally, we average the rates of the 10 rounds and calculate the standard deviation. The results are shown in the Figures 9, 10 and 11. In all cases, the model setting ([-1, 1], [-1, 1]) provides better or equivalent Coverage and Purity. However, the difference

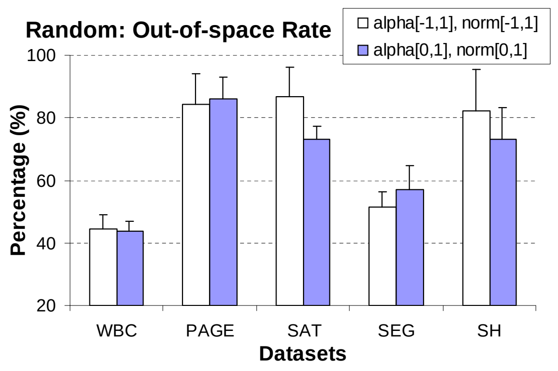

Figure 11: Comparison of Out-of-area rates in random rendering

of Out-of-area rate are not statistically significant (with p-value > 0.05).

Comparison on Visual Clustering Results. We also show some visualization results to give the intuition how the range settings affect the visualization results. The Figures, 15 vs. 14, 13 vs. 12 clearly contrast the effects of the range settings ([-1, 1], [-1, 1]) and ([0, 1], [0, 1]). The visualization results with the setting ([0, 1], [0,1]) are usually skewed and less detailed, as we have mentioned in the formal analysis.

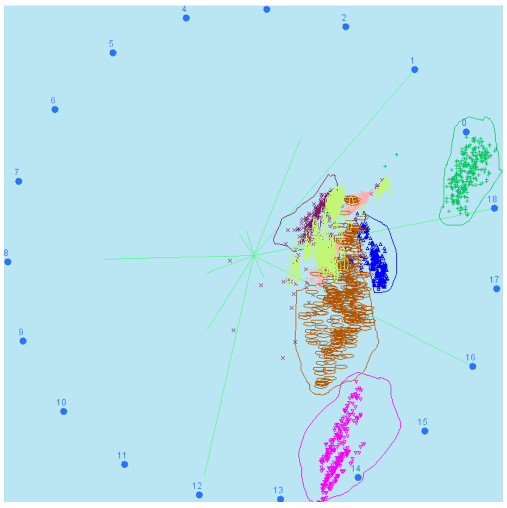

Figure 12: Sample visualization of SEG data with setting ([0, 1], [0, 1])

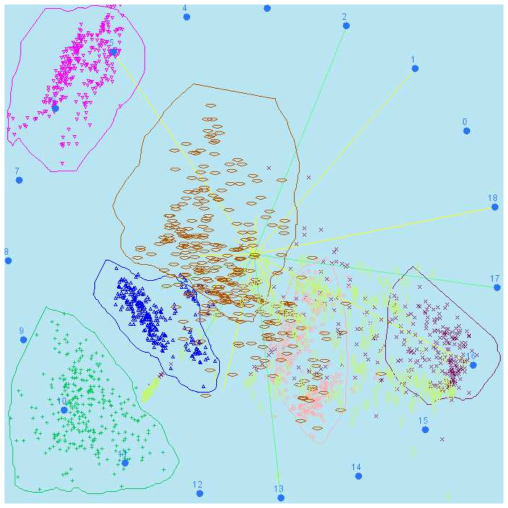

Figure 13: Sample visualization of SEG data with setting ([-1, 1], [-1, 1])

Better visualization results help find better clustering results. We try to identify the clusters based on the observed dense point clouds and manually label them. The results are compared with the existing cluster labels. The *Error Rate* of the result is calculated as follows. Suppose K

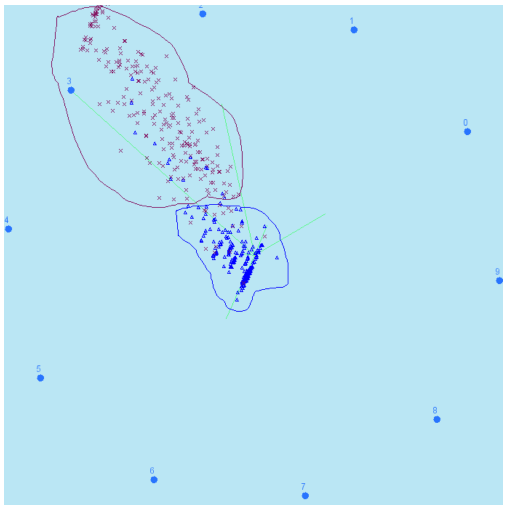

Figure 14: Sample visualization of WBC data with setting ([0, 1], [0, 1])

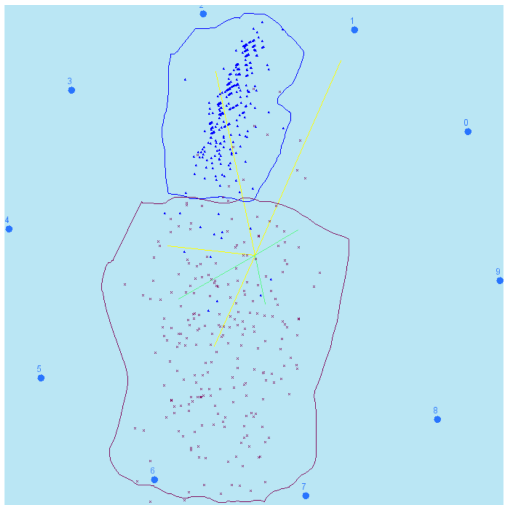

Figure 15: Sample visualization of WBC data with setting ([-1, 1], [-1, 1])

clusters are identified. Error Rate is defined based on the *confusion matrix*, where each element c_{ij} $1 \leqslant i, j \leqslant K$ represents the number of points from the originally labeled cluster j assigned to cluster i in the clustering result. Let {(1), (2), . . . , (K)} be a permutation of sequence {1, 2, . . . , K}. There is a permutation that best matches the visual clustering result and the originally labeled result that maximizes the number of consistent points m_c.

$$m_c = \max\{\sum_{i=1}^{K} c_{i(i)}, \text{for any } \{(1), (2), \ldots, (K)\}\}$$

We define the Error Rate as $1 - \frac{m_c}{N}$, where N is the total number of points. Our result shows the average accuracy (over five visual exploration results starting with random initial α parameter settings) increases from 91.6% to 95.7% for wisc-breast-cancer and from 67.2% to 75.2% for image-segmentation data, by using the setting ([-1,1],[-1,1]).

6 Conclusion

The star-coordinate based visualizations are effective in interactive visual clustering. The goal of interactions in such systems is to find the possible visual cluster overlapping caused by star-coordinate mapping models. To better understand the star-coordinate based cluster visualization, we unify the mathematical models of the existing systems and study the design of interactive operations, heuristic exploration rules, statistical properties of randomly generated visualizations, and the effect of parameter range settings. We also conduct a set of experiments to study the effect of range settings to the effectiveness of interaction and visualization. Our formal analysis and empirical study show the fundamental features of the star-coordinate visualization and indicate the optimal design principles for such systems.

References

[1] Ankerst, M., Breunig, M. M., Kriegel, H.-P., and Sander, J. OPTICS: Ordering points to identify the clustering structure. In *Proceedings of ACM SIGMOD Conference* (1999), pp. 49–60.

[2] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., andAndy Konwinski, R. K., Lee, G., Patterson, D., Rabkin, A., Stoica, I., and Zaharia, M. Above the clouds: A berkeley view of cloud computing. *Technical Report, University of Berkerley* (2009).

[3] Card, S. K., Mackinlay, J. D., and Shneiderman, B. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufman, 1999.

[4] Cayton, L., and Dasgupta, S. Robust euclidean embedding. In *Proceedings of International Conference on Machine Learning (ICML)* (New York, NY, USA, 2006), ACM, pp. 169–176.

[5] Chen, K., and Liu, L. VISTA: Validating and refining clusters via visualization. *Information Visualization 3*, 4 (2004), 257–270.

[6] Chen, K., and Liu, L. iVIBRATE: Interactive visualization based framework for clustering large datasets. *ACM Transactions on Information Systems 24*, 2 (2006), 245–292.

[7] Chen, K., Xu, H., Tian, F., and Guo, S. Cloudvista: Visual cluster exploration for extreme scale data in the cloud. In *SSDBM* (2011), pp. 332–350.

[8] Cleveland, W. S. Visualizing data. In *AT&T Bell Laboratories* (1993), Hobart Press.

[9] Cook, D., Buja, A., Cabrera, J., and Hurley, C. Grand tour and projection pursuit. *Journal of Computational and Graphical Statistics 23* (1995), 155–172.

[10] De Silva, V., and Tenenbaum, J. B. Global Versus Local Methods in Nonlinear Dimensionality Reduction. In *Advances in Neural Information Processing Systems 15* (2003), vol. 15, pp. 705–712.

[11] Draper, N. R., and Smith, H. *Applied Regression Analysis.* Wiley, 1998.

[12] (editor), M. J. *Learning in Graphical Models.* MIT press, 1998.

[13] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Second International Conference on Knowledge Discovery and Data Mining* (1996), pp. 226–231.

[14] Faloutsos, C., and Lin, K.-I. D. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proceedings of ACM SIGMOD Conference* (1995), pp. 163–174.

[15] Gallier, J. *Geometric Methods and Applications for Computer Science and Engineering.* Springer-Verlag, New York, 2000.

[16] Hinneburg, A., Keim, D. A., and Wawryniuk, M. Visual mining of high-dimensional data. In *IEEE Computer Graphics and Applications* (1999), pp. 1–8.

[17] Hoffman, P., Grinstein, G., Marx, K., Grosse, I., and Stanley, E. Dna visual and analytic data mining. In *IEEE Visualization* (1997), pp. 437–442.

[18] Inselberg, A. *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2009.

[19] Kandogan, E. Visualizing multi-dimensional clusters, trends, and outliers using star coordinates. In *Proceedings of ACM SIGKDD Conference* (2001), pp. 107–116.

[20] Ng, A. Y., Jordan, M. I., and Weiss, Y. On spectral clustering: Analysis and algorithm. In *Proceedings Of Neural Information Processing Systems (NIPS)* (2001).

[21] Shneiderman, B. Inventing discovery tools: Combining information visualization with data mining. *Information Visualization 1* (2002), 5–12.

[22] Teoh, S. T., and Ma, K.-L. Starclass: Interactive visual classification using star coordinates. In *Proceedings of SIAM International Conference on Data Mining (SDM)* (2003).

[23] Thomas, J. J., and Cook, K. A. *Illuminating the Path:The Research and Development Agenda for Visual Analytics.* the National Visualization and Analytics Center, 2005.

[24] Wang, J. T.-L., Wang, X., Shasha, D., and Zhang, K. Metricmap: an embedding technique for processing distance-based queries in metric spaces. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* (2005), 973–987.

[25] White, T. *Hadoop: The Definitive Guide*. O'Reilly Media, 2009.

[26] Yang, L. Interactive exploration of very large relational datasets through 3d dynamic projections. In *Proceedings of ACM SIGKDD Conference* (2000), pp. 236–243.