Validating and Refining Clusters via Visual Rendering

Keke Chen[1] Ling Liu[2]
College of Computing, Georgia Institute of Technology
801 Atlantic Dr.
Atlanta GA 30332
1-404-894-7008[1], 1-404-385-1167[2]
{kekechen, lingliu}@cc.gatech.edu

ABSTRACT

Clustering is an important technique for understanding and analysis of large multi-dimensional datasets in many scientific applications. Most of clustering research to date has been focused on developing automatic clustering algorithms or cluster validation methods. The automatic algorithms are known to work well in dealing with clusters of regular shapes, e.g. compact spherical shapes, but may incur higher error rates when dealing with arbitrarily shaped clusters. Although some efforts have been devoted to addressing the problem of skewed datasets, the problem of handling clusters with irregular shapes is still in its infancy, especially in terms of dimensionality of the datasets and the precision of the clustering results considered. Not surprisingly, the statistical indices works ineffective in validating clusters of irregular shapes, too. In this paper, we address the problem of cluster rendering of skewed datasets by introducing a series of visual rendering techniques and a visual framework (VISTA). A main idea of the VISTA approach is to capitalize on the power of visualization and interactive feedbacks to encourage domain experts to participate in the clustering revision and clustering validation process. The VISTA system has two unique features. First, it implements a linear and reliable mapping model to visualize *k*-dimensional data sets in a 2D star-coordinate space. Second, it provides a rich set of user-friendly and yet effective interactive rendering operations, allowing users to validate and interactively refine the cluster structure based on their visual experience as well their domain knowledge.

Keywords: Scientific Data Clustering, Information Visualization, Human Factor in Clustering

1. INTRODUCTION

Over the past decades most of the clustering research has been focused on automatic clustering algorithms. The automatic algorithms are known to work well in dealing with clusters of regular shapes, e.g. compact spherical shapes, but incur higher error rates when dealing with arbitrarily shaped clusters. Concretely, problems with the automatic clustering algorithms can be briefly summarized as follows:

- It is hard to handle arbitrarily shaped clusters, which are common in applications. Although, some new algorithms like CURE [3], WaveCluster [20] and DBSCAN [15], have addressed this problem and try to solve it in some situations, such as in low-dimensional datasets, or the shapes are elongated/enlarged regular ones. It is still considered as unsolved hard problems due to the complexity in multi-dimensional (>3D) space and the hard-predictable skewed cluster distribution.
- The arbitrarily shaped clusters also make the traditional statistical cluster validity indices ineffective [18]. For example, the compactness index of an elongated shape is not high but the quality of cluster is still considered as good.
- In the context of applications, some irregularly shaped clusters may be formed by combining two regular clusters or by splitting one large cluster with the incorporation of domain knowledge. However, it is inconvenient to incorporate domain knowledge in and allow the user to steer the clustering process with automatic algorithms.

One of characteristics of the automatic clustering algorithms is almost excluding human from the clustering process. What the user can do is setting the parameter before the clustering algorithm running, waiting for the algorithm producing the results, validating the results and repeating the entire process if the results are not satisfactory. Once the clustering algorithm starts running, the user cannot monitor or steer the cluster process, which also makes it hard to incorporate domain knowledge into the clustering process and inconvenient for large-scale clustering. This exclusion makes the existing clustering framework inefficient and unintuitive for the user to deal with application-specific clustering.

Visualization is the most intuitive to observe clusters, especially the clusters in irregular shape. For example, many clustering algorithms in literature employ the 2D-plot of the clustering results to validate the results on the 2D experimental datasets. The cluster visualization is not commonly used in practice because there is a difficult problem – preserving cluster structure in visualization for the multi-dimensional (>3D) datasets, keeping unsolved.

We propose a visual framework that allows the user to be more involved into the clustering process via interactive visualization. The core of the visual framework is the visual cluster rendering system VISTA. VISTA can start with any algorithmic results. At the beginning, VISTA imports the algorithmic clustering result into the visual cluster rendering system, and then lets the user to participate in following "clustering-analysis" iterations interactively. With the reliable mapping mechanism employed by VISTA system, the user can visually validate the defined clusters via interactive operations. The interactive operations also allow the user to refine the clusters or incorporate domain knowledge to produce better results.

VISTA system is able to project k-D datasets onto 2D Star coordinates [9] via an adjustable linear mapping – α-mapping. It does not break clusters but may cause cluster overlapping. The overlapping problem is solved by continuously tuning the visual parameters, mainly the α-parameter, which produces a series of continuously changed visualizations. The user is able to distinguish the cluster overlapping in the dynamically changed visualizations.

Combining with the algorithmic clustering results, VISTA works amazingly in improving the understanding of the cluster structure and the performance of validating and refining the arbitrarily shaped clusters. We will demonstrate the power of VISTA with two concrete examples – one is about how to validate and refine the algorithmic results with visual cluster rendering and the other is how to incorporate domain knowledge into the clustering process via visualization.

We organize the paper as following. The visual framework and VISTA system are introduced in section 2; in section 3, two empirical examples are demonstrated in details to show the power of VISTA in validating and refining clusters for real datasets. Then, the related work is discussed in section 4. Finally, we conclude our work and give some of the future work.

2. VISTA VISUAL FRAMEWORK

Most frequently, the clustering is not finished when the computer/algorithm finishes unless the user has evaluated, understood and accepted the patterns or results, therefore, the user has to be involved in the "clustering – analysis/evaluation" iteration. In many cases, a simplified process that employs automatic algorithms is like the following:

1. Run the algorithms with initial parameters.
2. Evaluate the cluster quality and analyse the clustering results with statistical measures and domain knowledge.
3. If the result is not satisfactory, adjust the parameters and re-run the clustering algorithms, then do step 2 again until the satisfactory result is found.
4. If the result is satisfactory, do post-processing, which may label the all of the items in the entire dataset or just output the cluster description.

Concrete discussion can be found in [14]. Our discussion will focus on steps 2 and 3. In step 2, it is often ineffective to validate the arbitrarily shaped clusters with the traditional cluster validity indices [18]. And it is also difficult for human to verify the result with the domain knowledge. In step 3, it is usually very time-consuming to find appropriate parameters for a new run. The user often has to try several sets of parameters and find the relations between the sets in the clustering results. For example, CURE [3] requires the parameter of the number of representative points and shrink factor and DBSCAN [15] needs a proper Eps and MinPts to get satisfactory clusters.

We observed that with automatic clustering algorithms steps 2 and 3 can only be done in sequence. The user can only tune the parameters before the algorithm running and then wait for the results and evaluate the results. We propose that if we can interweave these two steps, e.g. the user can participate in the clustering process, monitoring and steering the process, the entire process would be more efficient. Instead of achieving this interweaving by improving the existing automatic algorithms – which could be very hard – we develop an interactive cluster visual rendering system to get human involved in. The entire visual framework is like Figure 1.

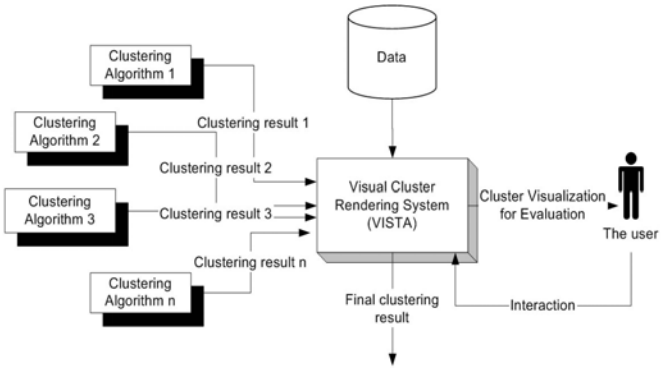

Figure 1. Visual framework for validating and refining clusters

Former studies [4] in the area of visual data exploration support the notion that visual exploration can help in cognition. Visual representations can be very powerful in revealing trends, highlighting outliers, showing clusters, and exposing gaps, especially interactive visualization. Previous research shows that, with the right coding, human pre-attentive perceptual skills can enable users to recognize patterns, spot outliers, identify gaps and find clusters in a few hundred milliseconds [17]. For example, in a scatter-plot based visualization, the human visual ability is adept at finding the clusters – the point-dense area very quickly, and the shape of the cluster is identified at the same time too. All of the advantages make the interactive cluster visualization systems very attractive.

However, there are some challenges for cluster visualization techniques, among which the most challenging one is cluster preserving– the clusters appearing in the 2D/3D visualization should be the real clusters in k-D (k>=3) space. Since a k-D to 2D/3D mapping inevitably introduces visual bias, such as broken clusters, overlapping clusters or fake clusters formed by outliers, static visualization is not sufficient and additional rendering techniques are needed to improve the visual quality.

In VISTA cluster rendering system, we use a linear (or affine) mapping [24] – α-mapping to avoid the breaking of clusters after mapping, but the overlapping and fake clusters may exist. The compensation technique is interactive dynamic visualization. The interactive operations are used to change the projection plane, which allows the user to observe the datasets from different angles. Continuously changed visualization usually provides important clues for the user to discriminate the overlapping and the fake clusters.

While the visual cluster rendering system is combined with the algorithmic result, the two can improve each other. The coloured algorithmic result in visualization provides visual clustering clues – the points in same colour, i.e. the same cluster, should be grouped in the same area, which can guide the user to find a satisfactory visualization. On the other side, the satisfactory cluster visualization after rendering can validate the

algorithmic results by visually checking the match of the visual cluster distribution and the algorithmic distribution. Therefore, the better way for visual cluster rendering is to combine the algorithmic results with the interactive visualization system.

The basic methodology employed in visual cluster validating and refining follows the steps:

Step1. Load the dataset, (and algorithmic result if available)

Step2. Use the interactive operations to find a satisfactory visualization,

Step3. Import domain knowledge if available, make the visual boundaries between clusters and refine the algorithmic result if applicable.

Step4. Output the refined result.

To illustrate how the VISTA works, we will briefly introduce the α-mapping and some interactive operations. The initial version of VISTA is used to render Euclidean datasets, where the similarity is defined by Euclidean distance, since the Euclidean datasets are the most common datasets in applications. By default, we will not mention this again in the following discussion.

α-mapping

We invent a linear mapping α-mapping that partially reserves k-d information in 2D space, and use it to build a *k*-parameter-adjustable interactive visualization system. The α-mapping model utilizes the form of 2D star coordinates [9] and normalizes the visualization into the designated display area.

A *k*-axis 2D star coordinates is defined by an origin $\tilde{o}$ (x_0, y_0) and *k* coordinates $S_1, S_2, ..., S_k$, which represent the *k* dimensions in 2D spaces. The *k* coordinates are equidistantly distributed on the circumference of the circle *C*, as in Figure 3, where the unit vectors are $\tilde{S}_i = (\hat{u}_{xi}, \hat{u}_{yi})$, *i*= 1..*k*, $\hat{u}_{xi} = \cos(2\pi / i), \hat{u}_{yi} = \sin(2\pi / i)$. The radius *c* of the circle *C* is constrained by the display area (e.g. <= width of display area/2).

α-mapping is a parameterized mapping that utilizes star coordinates to establish the visualization. We describe α-mapping as follows. Let a 2D point *Q* (*x*, *y*) represent a *k*-dimensional (*k*-d) max-min normalized [10] data point $P(x_0, x_1, ...x_i ..., x_k)$, $|x_i| \le 1$ in 2D star coordinates. *Q*(*x*, *y*) is determined by the average of the vector sum of *k* vectors $\tilde{S}_i \cdot x'_{ij}$ (*i*= 1..*k*) adjusted by *k* parameters (α_1, α_2,…, α_k) and scaled by the radius *c*.

α-mapping: $$A(x_1,...,x_k,\alpha_1,...,\alpha_k) = (c/k)\sum_{i=1}^{k}\alpha_i x_i \tilde{s}_i - \tilde{o} \quad (1)$$

i.e. $Q(x, y) =$

$$\left\{(c/k)\sum_{i=1}^{k}\alpha_i x_i \cos(2\pi/i) - x_0,\quad (c/k)\sum_{i=1}^{k}\alpha_i x_i \sin(2\pi/i) - y_0\right\},$$

The α_i (i = 1,2,…k, $-1 \le \alpha_i \le 1$) in the definition are dimension adjustment parameters, one for each of the *k* dimensions – that is where the name "α-mapping" comes from. In *Vista*, α_i is set to 0.5 initially.

The α-mapping has two properties:

- The mapping is linear. Without loss of generality, we set $\tilde{o}$ to (0, 0). It is easy to see the α-mapping is a linear mapping, given the constants α_i. It is known that the linear mapping does not break clusters but may cause overlapping clusters [24], and sometimes, overlapping outliers to form fake clusters. Given that the α-mapping is linear and thus there is no "broken clusters" in the visualization. All we need to do is to separate the overlapping clusters, and those falsely clustered outliers, which can be achieved with the help of interactive operations.
- The mapping is adjustable by α_i. The α_i (i = 1,2,…k, $-1 \le \alpha_i \le 1$) can be regarded as the weight of the *i*-th dimension, which means how significant the *i*-th dimension is in the visualization. By changing α_i continuously, we can see the effect of the *i*-th dimension on the cluster distribution. In addition, when one α value or several α values are changed continuously at the same time, the *k*-D dataset is mapped to a series of smoothly changed projections, which provide important cluster clues. In [1], we discussed and demonstrated that the dimension-by-dimension interactive exploration is also meaningful for efficiently exploring cluster distribution in Euclidean datasets.

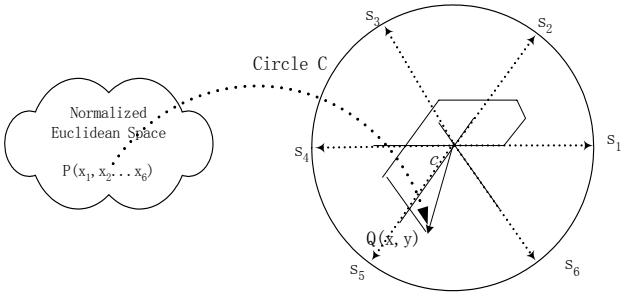

Figure 2. Illustration of α-mapping with *k*=6

The visual rendering operations

The VISTA system looks like Figure 2. The task of the VISTA cluster rendering system is to provide the interactive visualization techniques to help the users find and separate the overlapping clusters through continuously changed visualization. We have designed and implemented a set of interactive rendering operations in *Vista*. Due to the space limitation, we only introduce some of the operations. These operations can be formally described as set operations.

α-parameter adjustment

This operation changes the α parameters defined in formula (1). Each change refreshes the visualization in real time (about several hundred milliseconds). α-parameter adjustment enables the user to find the dominating

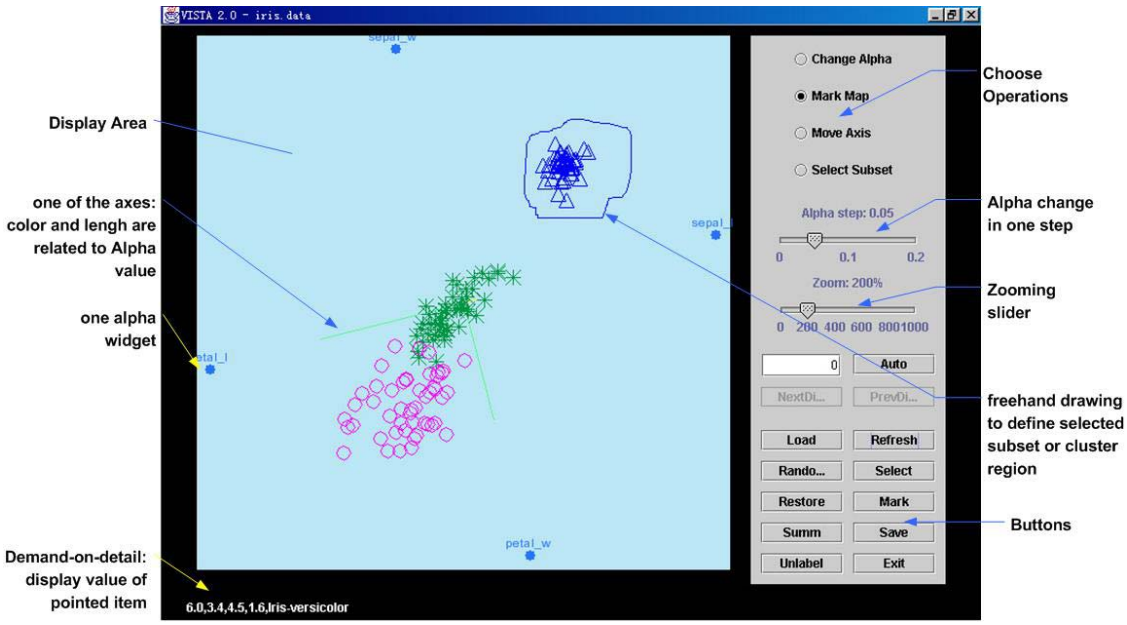

Figure 3. VISTA system

dimensions, to observe the dataset from different angles and to discriminate the real clusters from overlapping clusters with the continuously changed visualization. The user can use the operation to find basic skeleton of a cluster distribution. Random rendering and automatic rendering are another two automated α-parameter adjustment methods. **Random rendering** changes α parameters of all dimensions randomly at the same time and helps users find interesting patterns if the cluster distribution is not very obvious. **Automatic rendering** continuously changes the α parameter of one chosen dimension automatically. A user can switch to auto-rendering the next dimension or the previous dimension, or jump to any chosen dimension. This automatic rendering can save the user a lot of interactive operations.

α-parameter adjustment looks at the effect of one dimension to the entire visualization. Suppose we adjust the dimension *i*, then the point movement can be represented by:

$$\Delta(i) = \mathrm{A}(x_1,...,x_k,\alpha_1,...\alpha_i...,\alpha_k) - \mathrm{A}(x_1,...,x_k,\alpha_1,...\alpha_i'...,\alpha_k)$$
$$= (c/k)(\alpha_i - \alpha_i')x_i\tilde{s}_i$$

which means that the points having larger x_i will be moving faster, and the similar x_i moving in a similar way. This point movement reveals the characteristics of dimension *i*. Difference between x_i can reflected by the dynamic visualization. In Euclidean datasets, two points close to each other imply the values in each dimension are very close, which makes the dimension-by-dimension rendering very meaningful and effective in revealing clusters and overlapping.

Subset selection

This operation defines a subset by freehand drawing an enclosed region on screen or selecting a range of one dimension, which can be used for further processing, such as cluster marking, merging and dividing.

Initially, we have one subset, which is the entire dataset. The clusters are defined as subsets. We name *i*-th subset as ss_i. After loading labels, which define *c* clusters, the subsets becomes (ss_1, ss_2, …, ss_c). Suppose before selection, we have had *m* subsets ordered as (ss_1, ss_2, …, ss_m) . The (*m*+1)-th subset is selected from one or more subsets. We define subset selection as following, where '-' is set difference operation.

$$Select(m): (ss_1,...,ss_i...,ss_m) \rightarrow (ss_1 - ss_{m+1},...,ss_i - ss_{m+1}...,ss_m - ss_{m+1},ss_{m+1})$$

Merging & splitting clusters

These two operations enable the user to refine the visualized algorithmic clustering result. If the user finds a part of a cluster should be semantically separated from the cluster, she/he can use selection operation to select this part and then excludes it from the cluster. If two nearby clusters should be regarded as one cluster from the domain knowledge, the user just selects them and merges them into one cluster. A Cluster boundary can be refined by merging and dividing operations, too. Splitting subset *i* to subset *i*1 and *i*2, and merge subset i to j are defined as following, where 'U' is set union operation.

$$Split \quad (i,i1,i2,m): (ss_1,...,ss_i...,ss_m) \rightarrow (ss_1,...,ss_i - ss_{i2}...,ss_m,ss_{i2})$$
$$where \quad ss_{i1} = ss_i - ss_{i2}$$

$$Merge \quad (i,j,m):$$
$$(ss_1,..ss_i..ss_j...,ss_m) \rightarrow (ss_1,..ss_{i-1},ss_{i+1}...,ss_j \cup ss_i,...,ss_m)$$

Defining hierarchical cluster structure

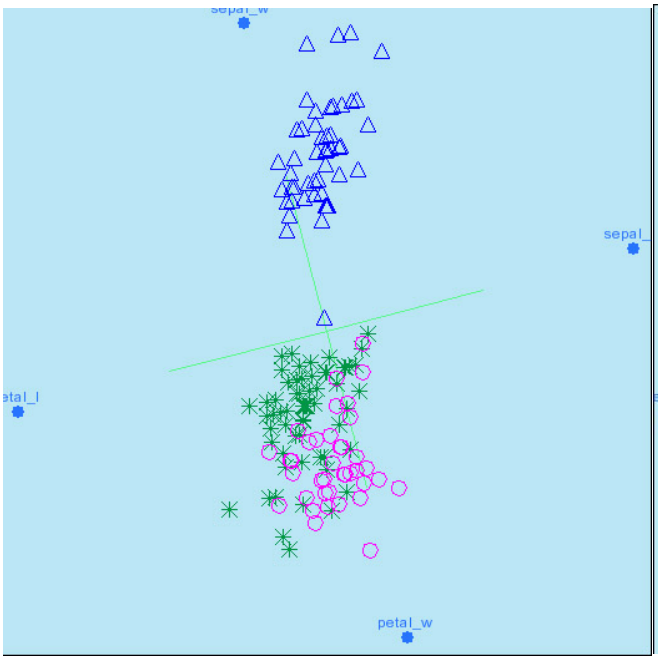

Figure 4: the initial visualization with k-means labels

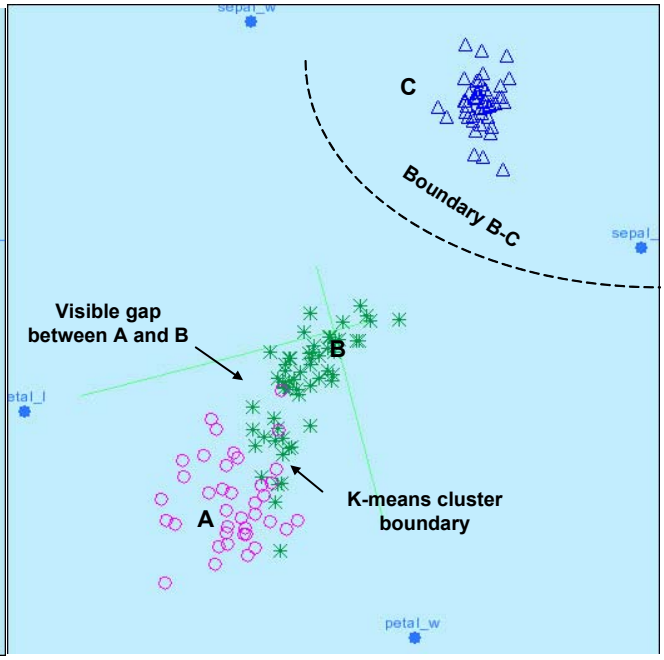

Figure 5: k-means result, RMSSTD =0.4421, RS = 0.8254, Error rate = 10.67%

With the operations of defining the cluster hierarchy, the user can group the clusters together to form a higher level cluster structure in layers. Or the user can zoom into one large cluster and find the fine cluster structure in the cluster iteratively. These operations define a layered cluster structure. With the operations of zooming in or zooming out, the user can find the cluster details at different level.

When we are rendering at some layer *j*, where it has *m* subsets, and want to group *k* selected clusters.

$$H_j(m, ss_{i1}, ss_{i2}..., ss_{ik}):$$

$$(ss_1, ss_2, ..., ss_m) \rightarrow (ss'_1, ss'_2, ..., ss'_{m-k}, (ss_{i1}, ss_{i2}..., ss_{ik}))$$

where, ss_j' is a ss_k ($k>=j$) before the layer operation. This operation can be recursively done layer by layer.

Importing domain knowledge

A set of domain knowledge is transferred to a set of k-D items with different group identities. These items are imported into the visual rendering system and rendered in different colours with different groups. These coloured items act as the guidance to re-define the cluster partition with domain knowledge.

If domain knowledge is represented by k groups of items, these items form *k* new subset after they are loaded.

$$D(m, g_1, g_2..., g_k):$$

$$(ss_1, ss_2, ..., ss_m) \rightarrow (ss_1, ss_2, ..., ss_m, g_1, ..., g_k)$$

3. EMPIRICAL STUDY

In the following section, we will introduce two examples of visual rendering clusters. The first one shows the ability of VISTA visually validating and interactively refining clusters. The second one shows how to incorporate domain knowledge into VISTA visual cluster rendering. The datasets used in the examples can be found at http://www.ics.uci.edu/~mlearn/Machine-Learning.html.

3.1 Analyzing the "Iris" dataset

In this example, we will use the most popular clustering algorithm – k-means[12] to produce the clustering result on the dataset "iris", and then import the result into VISTA system. With VISTA system, we will validate the k-means result visually and then try to refine the clusters and improve the quality of the k-means clusters. The quality of clusters will be also evaluated by one set of the widely used statistical indices RMSSTD (Root-Mean-Square Standard Deviation) and RS (R-Squared) [25][18] at the same time to see if the statistical indices are consistent with the visual improvement.

"Iris" dataset is a famous dataset widely used in pattern recognition and clustering. It is a 4-D dataset containing 150 instances, and there are three clusters, each has 50 instances. One cluster is linearly separable from the other two; the latter two are not exactly linearly separable from each other.

Firstly, we load the dataset and import the k-means labels for "iris" dataset into the visualization. Different clusters are visualized in different colors and shapes. The initial visualization is like Figure 4, where we can find one cluster has been separated from the other two. After interactive cluster rendering, mainly the α-parameter adjustment, the visual boundaries become clearer (Figure 5). The boundary B-C clearly separates cluster C from the other two clusters. The gap between cluster A and B can be visually perceived but not so clear. The α-mapping model confirms that this gap does exist in

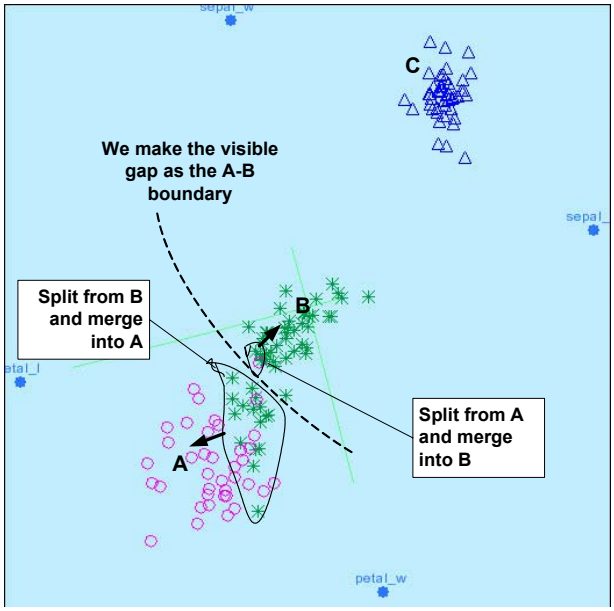

Figure 6: Editing the clusters

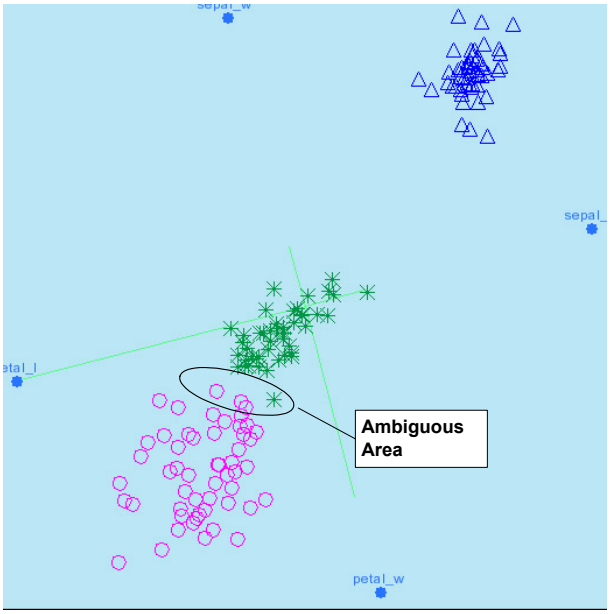

Figure 7: After editing, EMSSTD = 0.4614, RS=0.8098, Error rate = 2.67%

the 4-D space since α-mapping does not break clusters. We make this gap as the visual boundary A-B. This visually perceived boundary A-B is not consistent with the k-means boundary, but we have more confidence with it since it has been intuitively confirmed. There is a principle in visual cluster rendering – we prefer visual perception rather than statistical information because we believe the visual ability is better than statistical methods in dealing with arbitrarily shapes.

Considering this visual boundary, we want to edit the k-means result with visual cluster editing operations. First, we split the points that belong to cluster A but visualized in cluster B from cluster A. These points are then merged into cluster B. Do the same operation on the B points in cluster A as shown in Figure 6. After the editing operations, the points in the clusters are shown more homogeneously (Figure 7). The visual partition exactly reflects the real cluster distribution (compare Figure 7 and 8), and the error rate is dropped dramatically from 10.67% for k-means result to 2.67%.

We check the validating results of the widely used cluster validity indices RMSSTD and RS, to see if the statistical validation is consistent with the visual improvement. RMSSTD is used to estimate the homogeneity of the clusters. Smaller RMSSTD indicates that the clusters are more compact. RS is used to estimate the dissimilarity between clusters. Larger RS indicates higher dissimilarity between groups. The RMSSTD and RS are defined in [25].

The statistical evaluation shows RMSSTD is increased from 0.4421 to 0.4614, while RS is decreased from 0.8254 to 0.8098, which means the compactness of clusters and the dissimilarity between clusters are decreased at the same time – the quality of clustering after visual improvement is worse than the k-means result in statistics, which is not correct in practice! The irregular shapes of A and B, together with the closeness to each other, makes the statistical methods ineffective in this scenario.

As the literature of the "iris" dataset mentioned, the clusters A and B are not linearly separable. To further refine the cluster definition, we can also informally define a small "ambiguous area" around the gap between A and B, the points in which have equal probability of belonging to A or B.

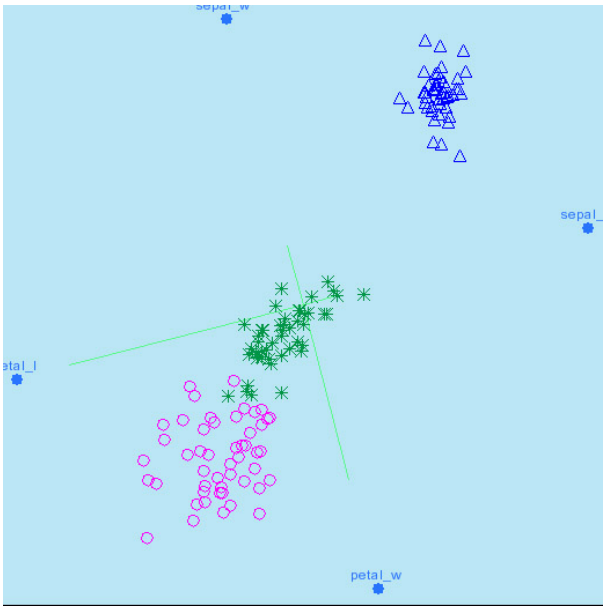

Figure 8: the real cluster distribution visualized with the labels from the original dataset.

In conclusion, we believe that the VISTA system is better than the statistical indices, in terms of validating arbitrarily shaped clusters. In this example, we have seen that sometimes the vague boundary between the two clusters is easily checked by human visual ability but it is not so easy for the automatic

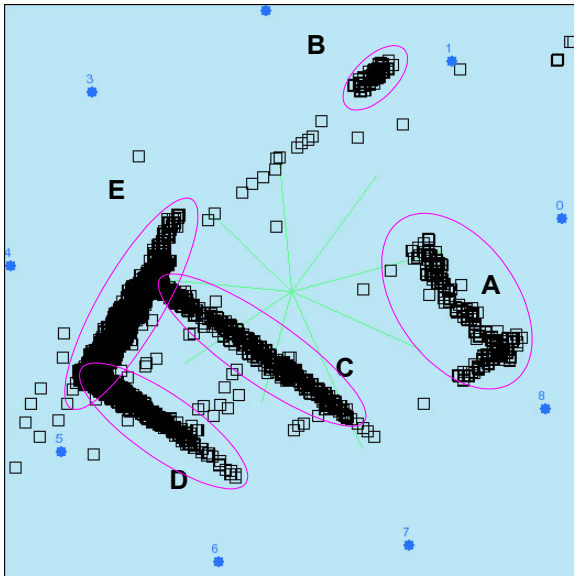

Figure 9: The visualization after initial rendering operations

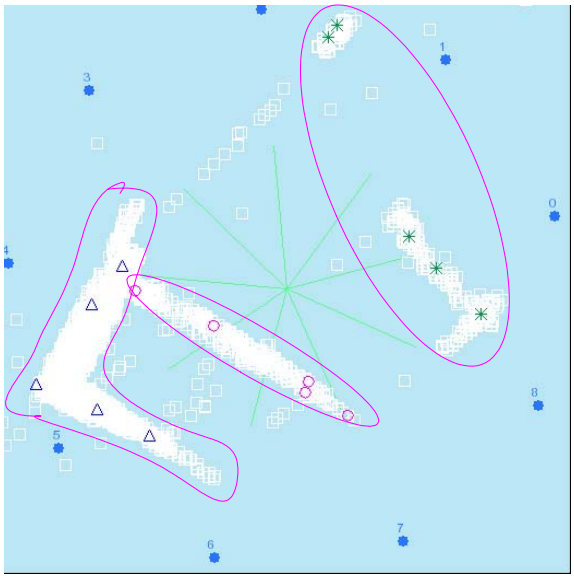

Figure 10: The landmarks and suggested cluster structure.

algorithms. In addition, this example also shows the power of online refining ability of the VISTA system – after validation, the user can improve the quality of clusters immediately by editing the clusters – which effectively combines the two steps "re-clustering" and "evaluation" together. Certainly, in cases where the clusters are not easily be visualized, e.g. clusters in very high-dimensional datasets, (e.g. >50 dims for VISTA), the statistical indices are still the only choice, even though it is not so effective.

3.2 Incorporating Domain knowledge

In this empirical example, we will demonstrate that the VISTA system can conveniently incorporate the domain knowledge into the clustering process and provide intuitive clues for the user to define the application-specific clusters. We first define the form of "domain knowledge" that can be utilized in VISTA system, and then show how to use the domain knowledge to distinguish the application-specific cluster distribution with the example of rendering "shuttle" dataset.

Domain knowledge plays a critical role in the clustering process [14]. It is the semantic explanation to the data, which is different from the structural clustering criteria, such as distance between points. Domain knowledge usually leads to a high-level cluster distribution, which may different from the structural clustering results, for example, the original clusters may be merged to form larger clusters or split to form finer cluster structure.

Domain knowledge can be represented in various forms in Artificial Intelligence [14]. However, in VISTA system, we need only one of the simplest forms to provide the domain-related clustering criteria. We define the domain knowledge as following:

Suppose the dataset contains a set of instances $\{X_i\}$ and the user have some knowledge about the application. The form of the domain knowledge can be the specific properties, the experimental results, or any hypotheses the application holds. We need a small number of typical instances $X_1, X_2, \ldots, X_n$ (n << the number of items N in the dataset) to reflect the properties, or the experimental results. According to the domain knowledge, this set of instances should be partitioned into *m* groups. The *m* groups are represented by $g_1(X_{1,1}, X_{1,2},\ldots, X_{1,t1})$, $g_2(X_{2,1}, X_{2,2}, \ldots, X_{2,t2}),\ldots g_m(X_{m,1}, X_{m,2}, \ldots, X_{m,tm})$. We give labels to the instances so that each instance is represented as (instance, label#). Therefore, we have the n instances labeled as

$(X_{1,1}, 1)\ (X_{1,2}, 1)\ldots\ (X_{1,t1}, 1)$

…

$(X_{m,1}, m)\ (X_{m,2}, m)\ldots\ (X_{m,mt}, m)$

They are regarded as additional points of the dataset, with domain categorical labels. We name them "landmarks" in VISTA system. The number of the instances is so small that they cannot work efficiently as a training dataset to do classification task on the entire datasets.

When visualizing a dataset, the landmark points are loaded and visualized in different colors according to their categorical ID. This guiding information can direct the user to define the high-level cluster structure, or to refine the algorithmic clustering results. Automatic algorithms have not such abilities, or it is very inefficient or clumsy to incorporate this functionality into the automatic algorithms.

We use the "shuttle" dataset to demonstrate how the VISTA system incorporates the domain knowledge into the clustering process. "Shuttle" dataset is a 9-D dataset. There are three large clusters and some tiny clusters in the dataset. Approximately 80% of the data belongs to one cluster. The other two large clusters have about 15% and 5% points, respectively. We use the testing dataset, which has 14500 items, for visualization.

After loading the dataset and adjusting the α parameters, we get a visualization, which shows the cluster distribution is highly irregular. There are five obvious segmentations (Figure 9), so totally we have C_5^3 - 10 possible combinations to form the 3

large clusters. Intuitively, the close clusters C, D and E are more likely to be defined as 1 or 2 clusters, but we are not sure yet.

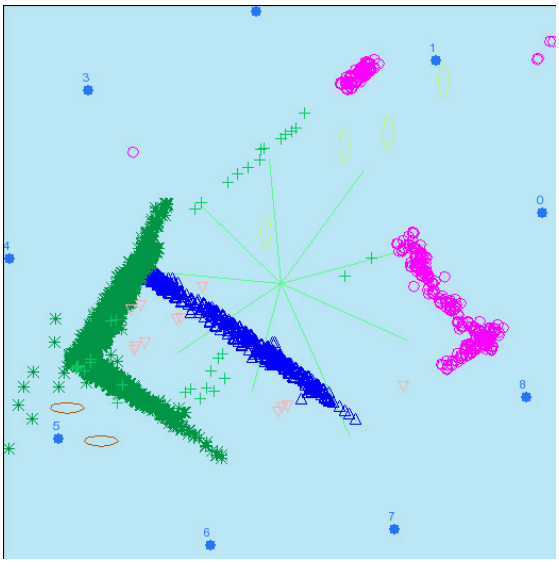

Figure 11: the clusters with original labels

Now we suppose we have known there are three large clusters. We pick some typical points at the "knots" in visualization from the labeled dataset, which are simulated as the real "landmarks". These landmarks are visualized in 3 colors according to their labels. To observe the landmarks clearly, we visualized other data points in white color. The result is Figure 10, which shows the datasets probably should be partitioned in the suggested way. The real cluster distribution of the "shuttle dataset" is visualized in Figure 10 for comparison.

To sum up, since the automatic algorithms exclude the human from the clustering process, the domain knowledge cannot be easily incorporated into the clustering process. With the help of VISTA system, the user is able to incorporate the domain knowledge into the clustering process to define application-specific cluster distribution online. This combination of human-based analysis/evaluation and clustering process breaks the gap between human and the machines, and thus improves the efficiency of the entire cluster analysis process.

3.3 More Experiment Results

The VISTA visual clustering system was implemented in Java. In this section we will introduce more experimental results to show the effective of combining visual cluster rendering and algorithmic result. These experiments were conducted on a number of well-known datasets that can be found in UCI machine learning database (http://www.ics.uci.edu/~mlearn/Machine-Learning.html). These datasets, although small in size, have irregular cluster distribution, which is an important factor for testing the effectiveness of the VISTA system.

When doing experiments, the categorical attributes in some datasets are simply mapped to integer numbers and then normalized to [-1, 1]. This simply normalization is actually works for some datasets like mushroom. After we use the interactive visual operations to find the satisfactory visualization, either solely by visual rendering or incorporated by algorithmic result, we mark the areas which is regarded as clusters and the items in each area is correspondingly labelled as this cluster. With the original labels in the datasets, we define the items that are wrongly clustered as the errors, the number of which divided by the size of the dataset is the error rate of visual cluster rendering on this dataset.

We first use unguided visual rendering (UGV) to find the visual partition first. Unguided visual rendering does not rely on any outside label information and only depends on the visually observed dense-point areas and the gaps between the areas. Since there is visual bias on the visualization, the visual rendering usually tends to trap in a local minima, where the user think the visualization is satisfactory for defining cluster boundaries. We wan to avoid this local minima by some outside algorithmic information. CURE clustering is recognized as one that can deal with irregular cluster shapes in some level. We then import the CURE clustering result as some algorithmic guiding information to see if this information can improve the UGV result. The experiment shows that individually CURE cannot deal the arbitrarily shaped clusters very well and UGV may trap into some local minima, but combining with CURE result (Comb) we can improve the UGV result more or less. The result also shows the visualization result, either UGV or combined rendering, is generally better than algorithmic result for arbitrarily shaped clusters.

Dataset	*N*	*k*	*n*	UGV	CURE	Comb
bre-canc-wisc	699	10	2	16.7	36.6	3.0
Crx	690	15	2	20.2	31.7	14.5
Iris	151	4	3	5.5	41.3	0.7
Page-blocks	5473	10	5	13.0	35.7	8.1
hepatitis	155	19	2	21.9	53.4	20.6
Heart	270	12	2	24.0	49.6	16.7
Mushroom	8124	21	2	24.7	36.8	2.5
Australian	690	14	2	15.4	35.7	14.4
Wine	178	12	3	7.9	34.3	3.4
Shuttle.test	14500	9	7	10.2	17.5	4.2

Table 3: Error rates of VISTA cluster rendering on typical datasets having irregular cluster distribution

We list a part of the experimental results in Table 3, where *N* is the number of rows in the given dataset, *k* is dimensionality of the dataset, and *n* is the number of clusters in the dataset. UGV is error rates (%) of unguided visual rendering result. CURE is error rates (%) of CURE clustering algorithm. 'Comb' is the error rates(%) of the combination of the UGV with CURE results as additional information. The result shows the visual cluster rendering combining with algorithmic result is pretty effective in finding satisfactory visualizations for the real datasets.

4. RELATED WORK

The common cluster analysis framework is described in the clustering review paper [14]. Recently, some algorithms have been developed to deal with arbitrarily shaped clusters. CURE [3] uses a set of representative points to describe the boundary of a cluster in its hierarchical algorithm. But the number of representative points increases dramatically with the increase of

the complexity of cluster shapes in order to maintain the precision. CHAMELEON [23] employs a multilevel graph partitioning algorithm on the k-Nearest Neighbour graph, which may produce better results than CURE on complex cluster shapes for spatial datasets. But the high complexity of the algorithms prevents its application on higher dimensional datasets. DBSCAN [15] is a density-based algorithm but it is very sensitive to the parameter Eps and MinPts. The distribution-based algorithm DBCLASD [22] and the wavelet transformation based algorithm WaveCluster [20] were also reported as being efficient only in spatial datasets. In conclusion, the automatic algorithms can deal with the arbitrarily shaped clusters in some situations, but the results are not general enough to apply to any application which has dimensionality higher than 3D. The most difficult problem is, for high-dimensional (>3D) datasets, the arbitrarily shaped clusters produced by the automatic algorithms are hard to be validated, since the statistical indices [18] are not effective for such clusters.

Information visualization is commonly recognized as a useful method for understanding sophistication in datasets. Many efforts have been made to analyze the datasets in a visual way. We discuss the scatterplot-based techniques only because it is the most intuitive techniques for cluster visualization. The early research on general plot-based data visualization is Grand Tour and Projection Pursuit [7]. 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 the interesting projections. L.Yang [8] utilizes the Grand Tour technique to show projections of datasets in an animation. They projected the dimensions to coordinates in a 3D space. However, when the 3D space is shown on a 2D screen, some axes may be overlapped by other axes, which make it hard to perform direct interactions on dimensions. Dhillon [5] provides a method for visualizing only 3 clusters while preserving the distances. When more than 3 clusters exist, his method needs the help of Grand Tour techniques. Other techniques, such as Scatterplot matrices, coplots, prosection [2] and FastMap based visualization [21, 19] only create static visualization, which inevitably distorts the cluster structure but have no effective methods to rectify it, thus do not provide enough information for correct clustering. In the KDD 2002 tutorial [13], some other visualization methods were also discussed.

Star Coordinates [9] is a visualization system designed to visualize and analyze the clusters interactively. We utilize the form of Star Coordinates and build a normalized α-mapping model in our system. The invention of α widgets also enables users to interact with visualization more efficiently. We also investigated the characteristics of dimension-by-dimension rendering in VISTA system [1].

5. CONCLUSION

Most of researchers have focused on automatic clustering algorithms, but very few have addressed the human factor in the clustering process. Recently, the existing clustering algorithms and cluster validity methods have encountered the difficulty in dealing with arbitrarily shaped clusters, which shows the limitation of the automatic approaches. In order to solve this problem, we probably should check the human factor in the clustering process more carefully.

In this paper, we tried to address and solve the limitation of the automatic approaches by getting the user more involved in the clustering process via visualization. For this purpose, we proposed a visual framework to combine the algorithmic results with visual cluster rendering system. The VISTA visual cluster rendering system provides reliable mapping mechanism to preserve the cluster structure partially, and effective interactive operations to help the user improve the cluster quality. The empirical study shows that the VISTA framework/system works very well in visually validating and refining algorithmic clustering results. Moreover, it also allows the user to incorporate domain knowledge into the clustering process in a convenient way.

The current VISTA system can handle datasets with dimensionality less then 50, dimensionality higher than or close to 50 which will cause the difficulty in human visual understanding and operations. In addition, the limitation of visualization system restricts the number of data items that can be handled. Currently, the VISTA system can handle about 50000 points and refresh the visualization in real-time (several hundreds of milliseconds). Therefore, the future work will be focused on handling high-dimensional and larger datasets.

REFERENCE

[1] Keke Chen and Ling Liu: "Cluster Rendering of Skewed Datasets via Visualization". ACM Symposium on Applied Computing 2003, Melborne, FL.

[2] W.S.Cleveland. "Visualizing Data", AT&T Bell Laboratories, Hobart Press, NJ.1993.

[3] G.Guha, R.Rastogi, and K.Shim. "CURE: An efficient clustering algorithm for large databases", in Proc. of the 1998 ACM SIGMOD

[4] J.Larkin and H.A.Simon. "Why a diagram is (sometimes) worth ten thousand words", Cognitive Science, 11(1): 65-99, 1987

[5] I. S. Dhillon, D. S. Modha& W. S. Spangler. "Visualizing Class Structure of Multidimensional Data", In Proc. of the 30th Symposium on the Interfaces, May 1998.

[6] D. Keim. "Visual Exploration of Large Data Sets", Communications of the ACM. August 2001, V. 44. No. 8

[7] Cook, D.R, Buja, A., Cabrea, J., and Hurley, H. "Grand tour and projection pursuit", Journal of Computational and Graphical Statistics, V23, pp. 225-250

[8] Li Yang. "Interactive Exploration of Very Large Relational Datasets through 3D Dynamic Projections", in Proc. of SIGKDD2000

[9] E. Kandogan. "Visualizing Multi-dimensional Clusters, Trends, and Outliers using Star Coordinates", in Proc. of SIGKDD2001.

[10] Liu, H. and Motoda, H. "Feature Extraction, Construction and Selection: A Data Mining Perspective", Kluwer Academic Publishers, Boston, 1998.

[11] Joseph B. Kruskal and Myron Wish. "Multidimensional scaling", SAGE publications, Beverly Hills,1978

[12] A. Jain and R.Dubes. "Algorithms for Clustering Data", Prentice hall, Englewood Cliffs, NJ, 1988

[13] Grinstein G., Ankerst M., Keim D.A.: Visual Data Mining: Background, Applications, and Drug Discovery Applications", Tutorial at ACM SIGKDD2002, Edmonton, Canada.

[14] Jain, A.K., Murty, M.N. and Flynn, P.J.: Data Clustering: A Review. ACM Computing Surveys, 31(3), P264-323
[15] Ester, M., Kriegel, H., Sander, J. and Xu, X. “A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”

[16] Hinneburg, A. and Keim, D. “ An Efficient Approach to Clustering in Large Multimedia Databases with Noise”, in Proc. of KDD-98, pp. 58-65
[17] Ben Shneiderman: Inventing Discovery Tools: Combinning Information Visualization With Data Mining”, Infromation Visualization 2002, 1, p5-12
[18] Maria Halkidi, Yannis Batistakis, Michalis Vazirgiannis: “Cluster Validity Methods: Part I&II”, SIGMOD Record, Vol31, No.2&3, 2002
[19] Zhexue Huang, David W.Cheung, Michael K. Ng: “An Empirical Study on the Visual Cluster Validation Method with FastMap”, Database Systems for Advanced Applications, DSFAA2001
[20]G.Sheikholeslami, S.Chatterjee, and A.Zhang. “Wavecluster: A multi-resolution clustering approach for very large spatial databases”, In Proc. VLDB98’, 1998
[21] C. Faloutsos, K. Lin, "FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Tradditional and Multimedia Datasets," in Proc. of SIGMOD 1995
[22] Xiaowei Xu, Martin Ester, H. Kriegel, J.Sander: “A Distribution-based Clustering Algorithm for Mining in Large Spatial Databases”
[23] G. Karypis, E. Han, V. Kumar: “CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modelling”, IEEE Computer, Vol32, No8, pp68-75, 1999
[24] Jean Gallier: “Geometric methods and applications: for computer science and engineering”, Springer-Verlag, NY, c2001
[25] Sharma, S.C.: “Applied Multivariate Techniques”, John Willy&Sons, 1996.