Measuring Data Abstraction Quality in Multiresolution Visualization [*]

Qingguang Cui[1,†], Matthew O. Ward[1], Elke A. Rundensteiner[1] and Jing Yang[2]
[1]Worcester Polytechnic Institute
[2]University of North Carolina at Charlotte

Abstract

Data abstraction techniques such as filtering, sampling, clustering, and summarizing can be used to reduce the size of a large dataset while maintaining the dominant characteristics of the original data. They are widely used in multiresolution visualization systems to reduce visual clutter and facilitate analysis from overview to detail. However, analysts are usually unaware of how well the abstracted data represent the original dataset, which can impact the reliability of results gleaned from the abstractions. In this paper, we define two data abstraction quality measures for computing the degree to which the abstraction conveys the original dataset: the Histogram Difference Measure and the Nearest Neighbor Measure. Each is inspired by information and abstraction measures that have been successfully used in other disciplines, including pattern recognition, image retrieval, image compression and approximate query processing. These measures have been integrated within XmdvTool, a public-domain multiresolution visualization system for multivariate data analysis that supports sampling as well as clustering to simplify data. Each abstraction quality measure is computed based on the data abstraction being displayed and presented to the analysts. These measures can be used to indicate the confidence level of the discovered patterns. Thus analysts can make more accurate decisions. Several interactive operations are provided, including adjusting the data abstraction level, changing selected regions, and setting the acceptable data abstraction quality level. Conducting these operations, analysts can select an optimal data abstraction level, trading off between the data density on the screen and data abstraction quality. Also, analysts can compare different abstraction methods using the measures to see how well relative data density and outliers are maintained, and then select an abstraction method that meets the requirement of their analytic tasks.

CR Categories: H.5.2 [Information Interfaces and Presentation]: User Interfaces—Graphical user interfaces I.5.3 [Pattern Recognition]: Clustering—Similarity Measures

Keywords: Metrics, Clustering, Sampling, Multiresolution Visualization.

1 Introduction

Very large multivariate datasets are increasingly common in many applications including bioinformatics, social science and data analysis for homeland security. To be effective, visualization tools must be increasingly capable of handling such huge datasets. As the number of data elements increases, two problems arise: displays become cluttered and response time deteriorates. Clutter saturates visualizations, obscures the structure of the visual display, and hinders visual data analysis. Increase in response time makes effective and efficient interactive exploration impossible.

Many abstraction techniques have been proposed to address this problem. They can be classified into two groups: data abstraction techniques in data space and clutter reduction techniques in visual space. Data abstraction techniques, such as filtering [1], clustering [10] and sampling [8], reduce data elements to be displayed and thus reduce clutter. Clutter reduction techniques assign more screen space to interesting data elements than to less interesting ones using zooming [3] or distortion [13].

Two of the most common data abstraction techniques are sampling and clustering. They effectively reduce clutter by displaying fewer data. However, we are faced with a new challenge when working with the abstracted data in place of the original data; it is unknown in many cases how well the selected abstraction represents the whole dataset and how reliable the patterns discovered based on this abstraction are. Typically no concrete criteria are available for analysts to compare different abstraction methods, and to select an abstraction level for a specific abstraction method. The measurement of data abstraction quality is one possible solution to resolve the above problems and should be an essential component of multiresolution visual analysis. Nevertheless, most prior work in the literature, including our own, do not use data abstraction quality measures to date. The one exception is the work by Bertini and Santucci [6], who present a quality measure for sampling and apply it for finding the optimum sampling level. However, this measure is limited to sampling. The authors do not consider other types or usages of abstraction measures.

In this paper, we develop two examples of abstraction quality measures, namely HDM (Histogram Difference Measure) and NNM (Nearest Neighbor Measure) and show how they can be utilized. Other data abstraction measures can be designed, such as measures based on statistical properties of the data. They can be applied to multiresolution visualization in the same way. The HDM is derived based on the average relative error [2] of aggregation used in approximate query processing of databases as well as image similarity measures [15, 23] used in image retrieval. The NNM is derived based on the nearest neighbor algorithm [9] used in pattern recognition and an image quality measure [18] used in image compression. We integrated these measures into several multivariate visualizations, including parallel coordinates, scatterplot matrices and glyphs, employing two dynamic bar charts to display the measures for the selected and the unselected regions of data. Several interactive operations have been designed for operating in this quality space, including adjusting the data abstraction level, changing the selected regions, regenerating the abstraction, and setting a desired quality level. Quality measures are recomputed whenever the above operations are performed. The measures and interactions together form an environment in which analysts can explore multiresolution visualization with abstraction quality information available.

Visual analysts can benefit from data abstraction quality measures in several ways. First, these measures give analysts a confidence level in the given abstraction they work with and thus also for any observation made based on the abstracted dataset. This enables them to make more accurate decisions. Second, these measures make analysts aware of the abstraction quality of the dataset. Better yet, interactive mechanisms are available for the analysts to control the abstraction quality. Thus they can find an appropriate abstraction level by trading off the accuracy of representing the data subset and the degree of visual clutter. Third, these measures can be used to compare the effectiveness of different abstraction methods. The analysts can thus select the abstraction method via a compromise

[*]This work was supported under NSF grants IIS-0119276 and IIS-00414380.

[†]E-mail:qgcui@cs.wpi.edu

between the relative data density, the degree to which outliers are preserved and response time. For some applications, it may be acceptable to use the abstracted datasets with moderate quality as long as the decisions can be rapidly made, while other applications may require a higher level of confidence in the data utilized.

The rest of the paper is organized as follows. We review related work in Section 2, and define two abstraction quality measures in Section 3. Sections 4 describes the integration of these measures with multiresolution visualization using sampling and clustering. Sections 5 and 6 present two case studies of the measures in use. Section 7 concludes this paper and discusses future work.

2 Related Work

Several researchers have proposed measures for visual quality, visual clutter and data abstraction. They have employed the measures to control interactive analysis on the visualization. Tufte [26] describes general measures for visual quality, such as the lie factor (the ratio between size of the effect shown in the graphic and size of effect in data) and the data density (the ratio between drawn data entries and the graph area). Bertini et. al. [5] give a model for measuring visual density and clutter in 2D scatterplots. They develop a clutter measure represented by the percentage of colliding pixels of all possible permutations. Peng et al. [16] propose visual clutter measures for parallel coordinates, scatterplots matrices, star glyphs and dimensional stacking. They use these measures to compute dimension orders with low visual clutter. Rosenholtz et al. [19] present a feature congestion measure for displaying clutter based on image retrieval techniques, evaluating similarity and correlation among different features in the visualization. This measure can be used in an automated user interface critiquing tool. The work of Bertini et al. [6] is closest to ours in that they have designed a quality measure for data abstraction. They partition 2D scatterplots into a grid of blocks, compare data densities of the original dataset and the data densities of samples in each block, and then calculate the percentage of matching blocks to achieve this measure. They use this quality measure to find an optimal sampling level. This measure is similar to the Histogram Difference Measure in our work, but the matching method of blocks is much coarser than our calculation of the histogram difference.

Sampling refers to the process of selecting and using subsets of observations to estimate some parameters about a population [25]; techniques include simple random sampling, stratified sampling and quota sampling. Sampling techniques have been well studied in statistics and widely applied in social science. In Computer Science, sampling is used for many tasks, including optimizing queries in databases with approximate information from samples [2, 24]. In recent years, faced with increasingly dense visualizations, researchers have begun to explore combining sampling with visualization. Dix and Ellis [8] demonstrate that random sampling can make the visualization of large datasets more perceptually effective. Their Astral Telescope Visualiser employs a 2D zooming interface to show data with different sampling levels. Bertini and Santucci [5, 6] employ a non-uniform sampling algorithm to select less data in dense areas to reduce clutter, and more data in sparse areas to maintain data characteristics. Rafiei and Curial [17] apply simple random sampling into network visualization and show that this sampling preserves the common characteristics of the network.

Clustering refers to the process of partitioning a dataset into groups of objects based on similarity between objects or proximity according to some distance measure [4]. Each group, called a cluster, consists of objects that are similar among themselves and dissimilar to objects in other groups. Clustering is an aggregation method, since a cluster is regarded as a higher level object that represents all objects it contains. It is widely used because of two reasons: 1) By visualizing cluster attributes rather than the original data, the number of visual elements displayed can be greatly reduced; 2) Clustering itself is a pattern discovering process. Thus visualizing clusters explicitly can reveal hidden patterns to viewers. Many visualization systems have adopted clustering methods to reduce clutter and analyze datasets. Fua et al. [10] cluster multivariate datasets, and navigate the hierarchy from clustering with a structure-based tool that supports drill-down, roll-up and brushing operations. They present a hierarchical version of parallel coordinates and later extend the work to other multivariate visualizations [27]. The InfoSky visual explorer [11] supports interactively exploring large, hierarchically structured document collections based on clustering. Kreuseler et al. [12] present a scalable framework for information visualization that can compute the clustering and hierarchy dynamically and support different methods to visualize clusters.

3 Quality Measures for Data Abstraction

Data abstraction is the process of reducing a large dataset into one of moderate size, while maintaining dominant characteristics of the original dataset. Some data abstraction methods select a subset of the original dataset as the abstraction, such as sampling and filtering, while other data abstraction methods construct a new more abstract representation, such as clustering and summarizing. Measurement generally refers to the process of estimating the magnitude of a quantitative property [7]. Measurement is essential for scientific research; with measurement, researchers can compare different objects and evaluate the effectiveness of programs or processes.

In this section, we will describe two abstraction quality measures in detail. To facilitate explanation of these measures, we define the **D**ata **A**bstraction **L**evel (DAL) as the ratio between the size of the abstracted dataset and the original dataset, and **D**ata **A**bstraction **Q**uality (DAQ) as the degree to which the abstracted datset represents original dataset. At a given DAL, the DAQ will vary based on the different abstraction methods used or even on different invocations of a given abstraction operation. A good abstraction method should maximize the data abstraction quality and minimize the variance of data abstraction quality in different invocations. The DAL can be considered as a very coarse data abstraction quality measure. Other data abstraction quality measures will, in general, be better descriptors than the DAL.

3.1 Histogram Difference Measure

A histogram is an aggregation method that conveys data distribution. To construct a histogram, the data space is partitioned into many small ranges, with each range corresponding to a bin. The height of a histogram bin is determined by the percentage of data points that fall in the corresponding range. It reveals the data density within each subrange.

We propose to use the difference between the normalized histograms of the original dataset and the abstracted dataset as a measure to gauge the DAQ. Let us assume that these two histograms have the same number of bins, and that bin sizes correspond to the percentage of the total number of data that fall in the bins. Bin difference is defined as the absolute difference between two bins. Then the histogram difference corresponds to the summation of bin differences between the corresponding bins in the two histograms. **HDM** (Histogram Difference Measure) is defined as the normalized histogram difference. Its range is from 0 to 1. 0 means in every pair of corresponding bins, at least one is empty and 1 indicates a perfect match. We express these with the following equations:

$$Pb_i = |Po_i - Ps_i| \tag{1}$$

where Po_i is the percentage of data that fall into the i-th bin of the original histogram, Ps_i is the percentage of data that fall into the i-th bin of the abstracted histogram, and Pb_i corresponds to their bin difference;

$$Ph = \sum_{i=1}^{N} Pb_i = \sum_{i=1}^{N} |Po_i - Ps_i| \tag{2}$$

where Ph is the histogram difference, and N is the number of bins.

$$HDM = 1.0 - \frac{Ph}{MAX_{Ph}} \tag{3}$$

where HDM is the Histogram Difference Measure, and MAX_{Ph} is the maximum histogram difference. These equations generate the HDM for one data dimension. The HDM for an N-dimensional dataset is defined as the average of the N 1D HDMs. Recall that the histogram represents a data distribution. Thus the proposed HDM represents the difference between the data distributions in the two datasets. If this difference is very small, the HDM will be near 1. In this case, the abstracted dataset represents the original dataset very well, implying the data abstraction method has a very high quality.

Thus far, we use the absolute difference between bins to calculate the histogram difference. If we consider a histogram as a vector, then we can treat the histogram difference as the distance between two vectors. Thus many general methods of calculating vector distances can be used to calculate the histogram difference. For example, the equation for Minkowski distance is:

$$Ph = \sum_{i=1}^{N} Pb_i = (\sum_{i=1}^{N} |Po_i - Ps_i|^p)^{\frac{1}{p}} \tag{4}$$

where p is the distance type, $p > 0$. For p=1, it is the Manhattan distance, which coincides with our definition of histogram difference above. For p=2, it is the Euclidean distance. We prefer to use the Manhattan distance as the histogram difference, because we feel it better conveys the difference in data distribution. The number of bins in a histogram influences its effectiveness in conveying information. We can compute the bin size by setting a bin width. The default bin width in our work is calculated using this equation: $W = 3.49S \times N^{-\frac{1}{3}}$, where S is the standard deviation in a given dimension and N is the number of data points. It has been illustrated by Scott [21] that this generally results in an effective bin size.

3.2 Validating Histogram Difference Measures

For very large databases, it may be prohibitively expensive to get exact results for aggregate queries. We note that the result of an aggregate query lists aggregate values in each sub-category. This in fact parallels a histogram. Error metrics are needed to measure the accuracy of the estimated query result compared to the actual query result. Babcock et al. [2] define the average relative error with the following equation:

$$RelErr = \frac{1}{n}(k + \sum_{j=1}^{m} \frac{|x_j - x'_j|}{x_j}) \tag{5}$$

where n is the number of bins, k is the number of empty bins in the estimated histogram, m is equal to $n-k$, x_j is the actual value of the j-th bin, and x'_j is the estimated value of the j-th bin. This metric is similar to our histogram-based measure except that it uses the percentage of each bin, while our measure uses the percentage of the whole dataset. With this error measure, they demonstrated that dynamic sampling can provide more accurate approximate results than non-adaptive usage of uniform or non-uniform sampling.

In the image retrieval field, features are extracted from images in order to facilitate searching over images. Image features are an abstraction of the image and often described as a histogram on image parameters such as color. Similarity measures are needed to compare histograms of image features. Swain and Ballard [23] first proposed the color indexing method, which compares the color histograms of two images, defined in Equation 6. Niblack et al. [15] proposed measuring the image similarity with the quadratic distance metric of a histogram using Equation 7.

$$Similarity(H_1, H_2) = \sum_{i=1}^{N} |H_{1i} - H_{2i}| \tag{6}$$

$$Similarity(H_1, H_2) = (H_1 - H_2)^R A(H_1 - H_2) \tag{7}$$

where H_1 and H_2 are two histograms, H_{1i} is the i-th bin of the first histogram, H_{2i} is the i-th bin of the second histogram, and A=[a_{ij}], where a_{ij} indicates the relationship between H_{1i} and H_{1j}. If all $a_{ij}(i \neq j) = 0$, then it becomes the Euclidean distance between the two histograms (Equation 8).

$$Similarity(H_1, H_2) = (\sum_{i=1}^{N} |H_{1i} - H_{2i}|^2)^{\frac{1}{2}} \tag{8}$$

Siggelkow [22] provided a comprehensive list of image similarity measures used in image retrieval systems. Our proposed histogram-based measure is similar to the first image similarity measure listed by Siggelkow, and is based on the same model as all similarity measures.

3.3 Nearest Neighbor Measure

As the name implies, the nearest neighbor algorithms [9] search for the object nearest to a given object. They are widely used to classify data into groups in data clustering and pattern recognition. Every object corresponds to a record. Each record in the original dataset has a nearest neighbor in the abstracted dataset, called its representative. The records in the original dataset, represented by the same record in the abstracted dataset, form a cluster. We define the **NNM** (Nearest Neighbor Measure) as the normalized average of distances between every record in the original dataset to its representative.

The following steps show the algorithm to calculate the NNM. 1) Choose a distance method to calculate the distance between two records. we use the Euclidean distance because it is the most common method for calculating distance between records. The equation is:

$$D(x, y) = \frac{\sqrt{\sum_{k=1}^{N} (x_k - y_k)^2}}{\sqrt{N}} \tag{9}$$

where x and y are two arbitrary records, $D(x, y)$ is the normalized distance between x and y, x_k and y_k are the k-th normalized dimension values of record x and y, respectively, ranging from 0 to 1, N is number of dimensions, and $\sqrt{N}$ is the maximum distance in the N dimension space. 2) Find a representative for each record in the original dataset. For the i-th record in the original dataset, we calculate the distances to all the records in the abstracted dataset, select the one with the smallest distance as the representative, and store this distance. The process is described with the following equation:

$$D_i = \min_{j=1}^{K} D(x_i, y_j) \tag{10}$$

where x_i is the i-th record in the original dataset, y_j is the j-th record in the abstracted dataset, K denotes the number of records in the abstracted dataset, and D_i is the distance of the i-th record to its representative. 3) Normalize the average of the minimum distances into the NNM. We use the following equation to do this:

$$NNM = 1.0 - \frac{\sum_{i=1}^{M} D_i}{M} \tag{11}$$

where D_i is the normalized distance of the i-th record to its representative, NNM is the Nearest Neighbor Measure, and M denotes the number of records.

3.4 Validating Nearest Neighbor Measure

The Nearest Neighbor Measure employs an algorithm to find a representative for each record, averages the normalized distances between data records and their representatives, and normalizes this value. To support the use of the Nearest Neighbor Measure, we first show that the nearest neighbor algorithm has been successfully used in pattern recognition, and then show that some image quality measures in image compression are derived from the average distance between pairs of image pixels in a method similar to our measure.

In pattern recognition, the nearest neighbor algorithm is used to classify phenomena based on observed features [9]. Phenomena and features are described in a vector. In the training stage, feature vectors are extracted from a set of observed objects. In the testing stage, a vector is extracted from a new phenomena, and the distances from this vector to all feature vectors are computed. Then the feature vector with the smallest distance is the nearest neighbor. This phenomena is assigned to the class that its nearest neighbor belongs to. We use the same algorithm to find the representative for each data record.

Image quality measure is essential for image compression. It is not only used to evaluate the compression technique, but also to control the compression process and decide how many bits are allocated to each subband [18]. Many image quality measures can be derived from the total or average distance between pairs of image pixels. The PSNR (peak signal-to-noise ratio) is the most common image quality measure, derived from MSE (mean squared error) and used in the JPEG 2000 Standard [20]. It is defined by the following equations:

$$MSE = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}(F(i,j) - \hat{F}(i,j))^2}{NM} \tag{12}$$

where MSE is the mean squared error, $F(i,j)$ is the pixel value at (i,j) in the original image, $\hat{F}(i,j)$ is the pixel value at (i,j) in the compressed image, and M and N are the length and height of the image.

$$PSBR = 10\log_{10}(\frac{MAX_I^2}{MSE}) \tag{13}$$

where $PSBR$ is the peak signal-to-noise ratio and MAX_I is the maximum pixel value. As we can see, the NNM employs the same method to compute the average distance between two datasets. The only difference is that they employ different methods to process the average distance to get the measures.

4 Integrating Quality Measures with Multiresolution Visualization

In this section, we describe our work on integrating quality measures into XmdvTool to develop effective and abstraction-aware multiresolution visualization. First we describe the interaction tool that we use to display quality measures. Then we present the interactive operations we support for quality measures. Next, we discuss the view continuity problem of sampling, and finally we give an overview of the Structure-Based Brush (SBB) we use to control abstraction parameters in clustering data. Analysts can adjust the DAL of clustering through both the general widget for all abstraction methods and the SBB, while they can only brush the structure formed by clustering through the SBB.

4.1 Displaying Measures

XmdvTool supports interactive selection via brushing [10, 14] using a rich assortment of tools. The data selected through brushing is called the selected data, while other data are called the unselected data. Analysts can adjust the DAL for the selected data as well as the unselected data. Each view of the data generates several quality measures. We use bar charts to display them. Figure 1 shows two such bar charts, the left one conveys the quality measures for the selected data, and the right one conveys the quality measures for the unselected data.

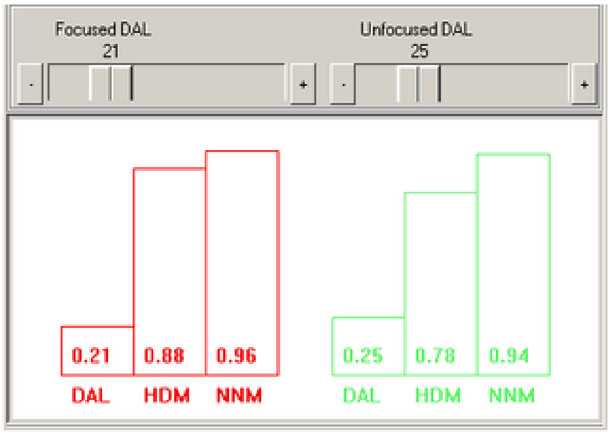

Figure 1: Graphs to display measures and sliding bars to adjust the DAL

These charts only illustrate the quality measures at a single DAL. We use 1D plots to illustrate quality measures and their relationship to DAL. In Figure 2, the left and right plot show the quality measures for the selected and unselected regions, respectively. In each plot, the x-axis represents the DAL and the y-axis represents the quality measures. The red and blue line represent the changes of HDM and NNM against the abstraction level, respectively. A vertical line called the DAL handle is drawn to indicate the current abstraction level. The cross points of this vertical line and the plot lines denote the corresponding measures of this abstraction level. The DAL and measures are displayed to the right of the DAL handle.

4.2 Interactive Operations

Several interactive operations are supported in this system. Users can move the slider bar in Figure 1 or the DAL handle in Figure 2 to adjust the data abstraction level. After the DAL has been changed, the system will generate an abstracted dataset and display it in the data visualization. The DALs for selected and unselected data can be adjusted independently. Users can also modify the location of one of the boundaries of the selected selected region by clicking the left mouse button on or near the boundary and dragging in the desired direction. In addition, the selected region can be moved by choosing a region on the data display, and then adjusting the DAL for the region. This usually means that the user knows the data subset that she wants to explore and wants to take advantage of the scalability of multiresolution visualization. Alternatively a user can subtree in more or less detail. Figure 3 shows the widget to con-

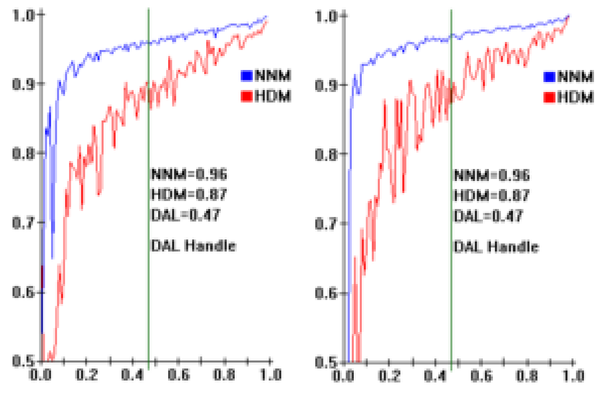

Figure 2: 1D plots of quality measures

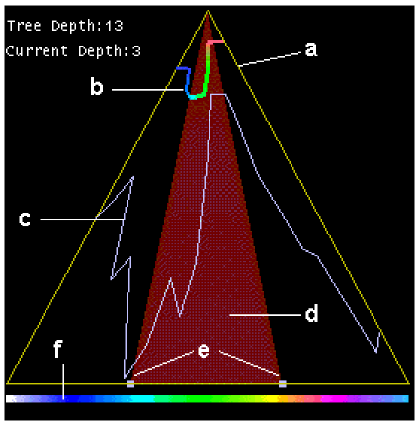

Figure 3: Structure-based brushing tool. (a) The tree frame; (b) Contour corresponding to current level-of-detail; (c) Leaf contour approximates shape of the tree; (d) Structure-based brush; (e) Interactive brush handles; (f) Colormap legend for level-of-detail contour.

first choose a DAL in the current selected region, and then adjust the selected/brushing boundary to enlarge or diminish the size of the region. This usually means that an acceptable data abstraction level had been found, but the area of interested needed to be increased or decreased. Analysts can also instruct the system to run the abstraction algorithm again to generate a new abstraction. For example, resampling can help analysts verify the pattern that had been discovered in the previous samples. If the pattern still exists after resampling several times, this pattern is most likely a robust one. Finally, a user can indicate a desired quality level based on one of measures and let the system decide the appropriate DAL.

4.3 View Continuity for Sampling

When analysts change the DAL, the patterns in the previous sample can be more easily remembered and compared with those in the current sample if the view continuity is maintained. This can be accomplished by following the three guidelines below: 1) When analysts change from a low DAL to a higher DAL, all the records in the previous sample should be kept. 2) When analysts change from a high DAL to lower DAL, all the records in the new sample should come from the previous sample. 3) When analysts broaden or the narrow the brushing boundary, the system should keep the records from the previous view, and then employ the same rules as above. We follow the above guidelines to maintain the view continuity. Analysts still have the option to resample at any time or whenever they change the DAL or the selected region.

4.4 Widget to Control Cluster-Based Abstraction

Hierarchical clustering generates a tree of clusters ranging from a single cluster containing the entire dataset to terminal clusters containing one record each. To represent a cluster in multiresolution visualization, one member of the cluster can be selected as a representative or a new record can be constructed to summarize the records in this cluster. This new record becomes the parent of all the records or clusters it contains. By recursively clustering data into related groups, a tree of clusters is formed. If the tree is visited using an in-order traversal algorithm, then all the nodes of this tree can be sorted and each node corresponds to a unique position in this order. Brushing is thus achieved via adjusting the range and selecting a subtree whose nodes fall in the range of this order. All the selected nodes will be displayed in the data visualizations. Analysts can adjust the abstraction level and visualize the data in this

trol both the level of abstraction and brushing, referred to as the Structure-Based Brush (SBB) [10]. The triangular frame depicts the tree (see (a)). The leaf contour (see (c)) depicts the silhouette of the tree. It delineates the approximate shape formed by chaining the leaf nodes. The colored bold contour (see (b)) across the tree delineates the tree cut that represents the abstracted dataset in a specific data abstraction level. Analysts can adjust the DAL by moving this contour. The two movable handles (see (e)) on the base of the triangle are called range handles. The range handles, together with the apex of the triangle, form the selected region in the structure space (see (d)). Analysts can adjust the selected region by moving the left handle, the right handle or both. This interface, while specific to hierachically clustered data, can support all of the interactions on the abstraction.

5 Case Study 1: Choosing a Data Abstraction Level (DAL)

In this section, we show how to choose an appropriate DAL. At this level, the abstracted dataset should have high data abstraction quality (equal or more than 0.90) and the visualization should have the best visual quality under the constraints of the data abstraction quality. The analytic task is to search for clusters in the OUT5D dataset. This dataset consists of five remote sensing channels: SPOT, Magnetics, Potassium, Thorium and Uranium, with 16384 records. We employ scatterplots to visualize this dataset. Figure 4 shows the original dataset. Data points have significant overlaps with each other and so we cannot distinguish relative data density in different regions and have difficulty observing any trends within this dataset.

First we make an abstraction with the DAL equal to 0.02. The corresponding HDM is 0.92 and the NNM is 0.93, and this abstraction quality meets our requirements. The abstraction quality is positively related to data abstraction level in general, although small fluctuations may exist. Thus the abstraction quality will meet our requirement in most cases as we move up the DAL. The scatterplot matrix with DAL equal to 0.02 is shown in Figure 5; we can see that a cluster (named as Cluster A) exists in the marked scatterplot, but data points in other places are too sparse to observe definitive clustering behavior. Next we will focus on searching for a visualization with the best visual quality.

We change the DAL to 0.08. As shown in Figure 6, the sparse region in the marked scatterplot illustrates very good visual quality.

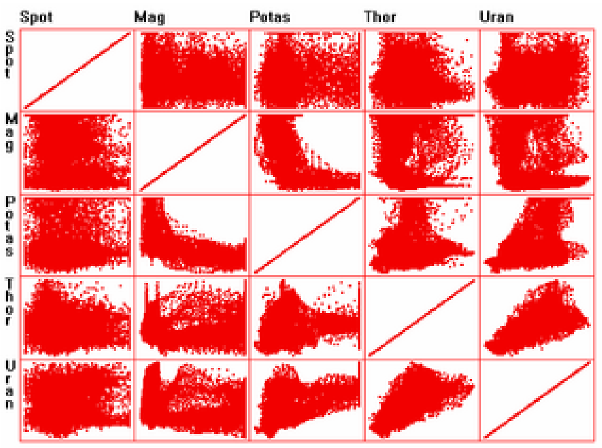

Figure 4: Scatterplots of original dataset (DAL=1.00)

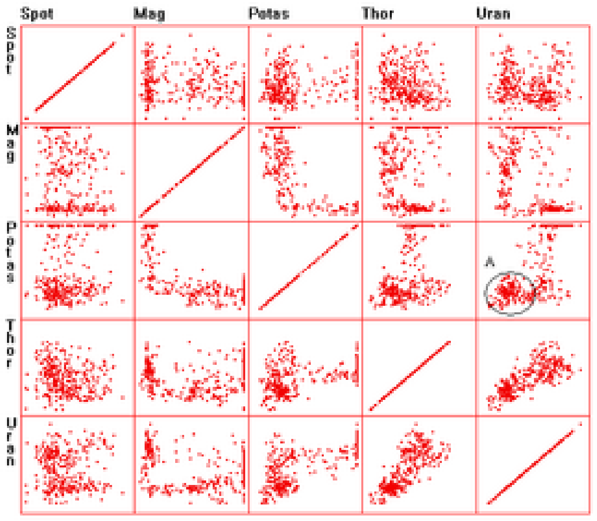

Figure 5: Scatterplots of abstracted dataset (DAL=0.02)

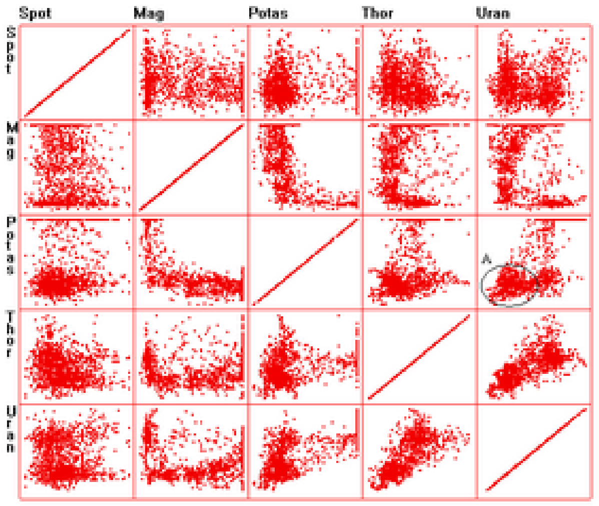

Figure 6: Scatterplots of abstracted dataset (DAL=0.08)

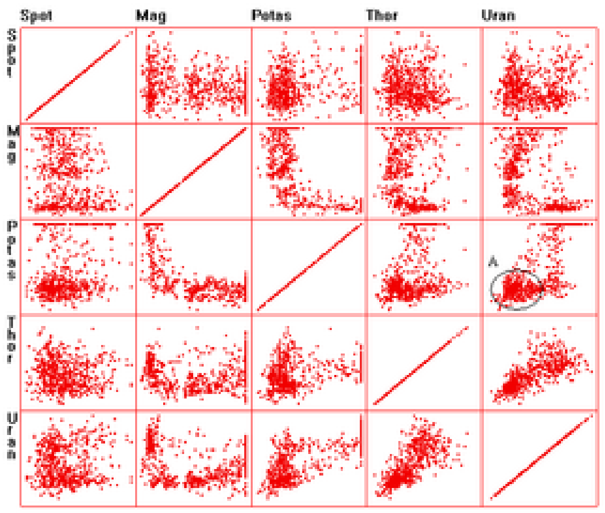

Figure 7: Scatterplots of abstracted dataset (DAL=0.04)

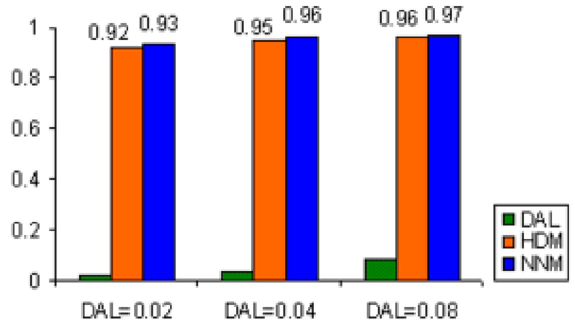

Figure 8: Data abstraction measures

However, data points in Cluster A are overplotted, thus the actual relative data density in Cluster A is higher than the relative data density we observe. Next we adjust the DAL to 0.04. As shown in Figure 7, the visual quality in the marked scatterplot is very good while the relative data density are maintained, although a small number of data points in Cluster A still overlap with each other. The quality measures of this abstraction are shown in Figure 8. This quality meets our requirement and we terminate our exploration.

Abstraction quality measures give us confidence in the pattern we discovered. If we only know that the DAL, the ratio between the number of abstracted records and the number of original records, is 0.04, we cannot have much confidence in our discoveries because we know that 96 percent of the data are not shown. However, with the HDM more than 0.95 and the NNM more than 0.96 for both clustering and sampling, we are fairly certain that the abstracted dataset represents the original dataset very well and that the pattern (Cluster A in this case) is very likely valid. In general, we can assign the abstraction quality measures to the discovered pattern to indicate the confidence level of the pattern, which enables analysts to make more accurate decisions.

6 Case Study 2: Comparing Data Abstraction Methods

In this application, two data abstraction methods, clustering and sampling, are compared using the proposed data abstraction measures embedded within our multiresolution visualization system. We employ the AAUP dataset, which surveys the number, salary and compensation of professors at 1161 institutions. We use parallel coordinates to visualize this dataset. Through this case study, we find that sampling has the advantage of maintaining the relative density of datasets while clustering has the advantage of maintaining the outliers of the dataset.

First we briefly review some characteristics of the HDM and NNM. The HDM is based on the histogram and minimizes the difference between the distributions of two datasets, so it excels in detecting changes in the relative density of data. The NNM minimizes the distance between the original dataset and the abstracted dataset. Outliers cannot be eliminated during abstraction without the increase of the average distance, because they tend to be far from most of the data records. Thus the NNM method gives high priority to outliers and is good at monitoring the change of outliers.

The original dataset is shown in Figure 9. On the last dimen-

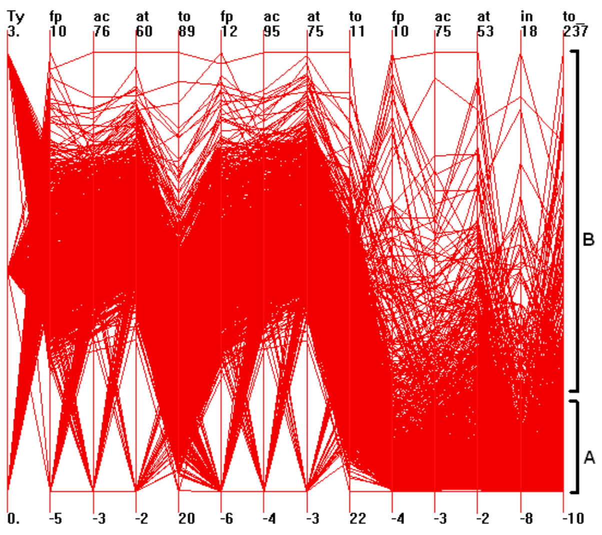

Figure 9: Parallel Coordinates of AAUP dataset

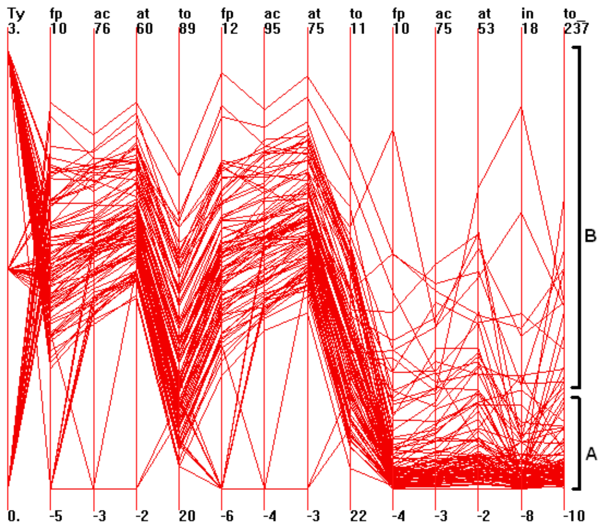

Figure 10: Parallel Coordinates of sampled AAUP dataset (DAL=0.08)

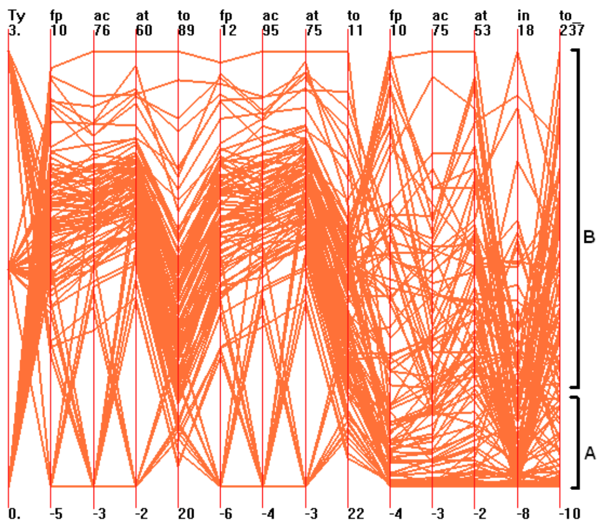

Figure 11: Parallel Coordinates of clustered AAUP dataset (DAL=0.08)

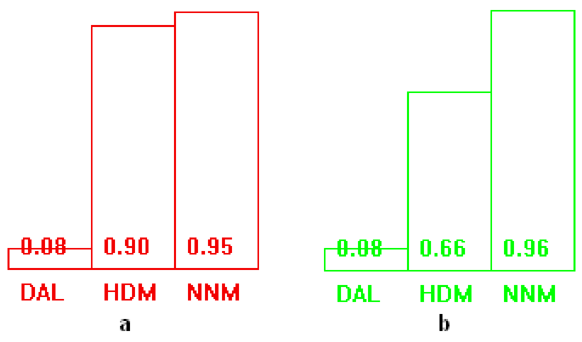

Figure 12: a. Quality measures for the abstraction from sampling; b. Quality measures for the abstraction from clustering

sion, the dense range with low values is marked as A and the sparse range is marked as B. We can see that most of the data records are gathered in range A. We sample the original dataset and tentatively set the DAL to 0.08. Figure 10 shows the visualization of this abstraction. Figure 12a shows the data abstraction quality: HDM is 0.90 and NNM is 0.95. We then cluster this dataset, and also set the DAL to 0.08 to facilitate comparing. As displayed in Figure 11, the visual clutter is significantly reduced. Figure 12b shows the data abstraction quality: HDM is 0.66 and NNM is 0.96.

We can see that the HDM of sampling is much better than the HDM of clustering. Thus we conclude that sampling maintains the relative density of the dataset much better than clustering. This can be explained by the fact that clustering finds one representative for each cluster, no matter how many members the cluster represents. Thus it loses the relative density. This can be verified by the visualizations. We can clearly see that sample-based visualization maintains the relative density, while cluster-based visualization reduces the relative density in range A and enlarges it in range B. On the other hand, the NNM is slightly better than that for sampling. We test sampling and clustering at other abstraction levels and get similiar results. So we can say that clustering maintains the outliers a little better than sampling. Analysts can consider the importance of maintaining relative density versus outliers for their analytic tasks, observe the HDM or NNM measures, and then select an abstraction method that balances relative density and outliers to meet their goals.

7 Conclusion and Future Work

In this paper, we have identified data abstraction as a common mechanism for dealing with large-scale data visualization. We designed two data abstraction quality measures to gauge how well the abstracted dataset represents the original dataset: the Histogram Difference Measure (HDM) and the Nearest Neighbor Measure (NNM). We implemented these measures within XmdvTool, which supports both sampling and clustering as abstraction methods. Several interactive operations were developed, including adjusting the data abstraction level, changing selected regions, regenerating the abstraction, and setting the desirable abstraction quality. The quality measures indicates the quality of the abstraction and thus also indirectly the quality of any patterns discovered through the abstraction.

Aided by these measures, analysts can find the most appropriate data abstraction level for a given task, that is, one with a reasonable abstraction quality and acceptable data density. These measures can also be used to compare different data abstraction methods in terms of how well they maintain relative data density and outliers. Thus our framework enables analysts to select abstraction methods that best fit their analytic tasks. We provide two case studies to illustrate the usefulness of these measures and the effectiveness of our proposed interactive tools related to the measures.

Ideally, the data abstraction quality measures should conform to the data abstraction quality as perceived by analysts. As shown in Section 3, many alternative measures could be devised beyond HDM and NNM, including even variations for computing the HDM and NNM. In our future work we will compare and evaluate how well the different realizations of HDM and NNM agree with the quality perceived by analysts. We will also explore ideas for new data abstraction measures based on statistical properties of the data, such as mean value and standard deviation. Different abstraction measures may be sensitive to the changes in different dataset features, such as the relative data density and characteristics of outliers. So we will also evaluate the advantages and limitations of abstraction measures in the presence of the dataset features. Peng et al. [16] designed and implemented several clutter quality measures based on visual clutter into XmdvTool, and we are also integrating data quality attributes within our visualizations. We plan to integrate these three efforts at quality measurement, representation and optimization and provide a quality-aware visualization framework to support the visual analysis process. Through this framework, analysts will be able to interactively explore very large datasets with quality information and refinement techniques available at each stage of the visualization process.

References

[1] C. Ahlberg and B. Shneiderman. Visual information seeking using the filmfinder. *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems*, 2:433, 1994.

[2] B. Babcock, S. Chaudhuri, and G. Das. Dynamic sample selection for approximate query processing. *Proc. ACM SIGMOD International Conference on Management of Data*, pages 539–550, 2003.

[3] B. Bederson. Pad++: Advances in multiscale interfaces. *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems*, page 315, 1994.

[4] P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002.

[5] E. Bertini and G. Santucci. By chance is not enough: preserving relative density through nonuniform sampling. *Eighth International Conference on Information Visualisation (IV'04)*, pages 622–629, 2004.

[6] E. Bertini and G. Santucci. Quality metrics for 2d scatterplot graphics: automatically reducing visual clutter. *Proc. of 4th International Symposium on SmartGraphics*, pages 77–89, 2004.

[7] J.L. Bucher. *The Metrology Handbook*. ASQ Quality Press, Milwaukee, Wisc., 2004.

[8] A. Dix and G. Ellis. By chance - enhancing interaction with large data sets through statistical sampling. *Proc. Advanced Visual Interfaces*, pages 167–176, 2002.

[9] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern classification*. John Wiley and Sons, Inc., 2th edition, 2001.

[10] Y. Fua, M. Ward, and E. Rundensteiner. Structure-based brushes: A mechanism for navigating hierarchically organized data and information spaces. *IEEE Trans. on Visualization and Computer Graphics*, 6(2):150–159, 2000.

[11] M. Granitzer, W. Kienreich, V. Sabol, K. Andrews, and W. Klieber. Evaluating a system for interactive exploration of large, hierarchically structured document repositories. *Proc. IEEE Symposium on Information Visualization*, pages 127–134, 2004.

[12] M. Kreuseler, N. Lopez, and H. Schumann. A scalable framework for information visualization. *Proc. IEEE Symposium on Information Visualization*, pages 27–36, 2000.

[13] Y. Leung and M. Apperley. A review and taxonomy of distortion-oriented presentation techniques. *ACM Transactions on Computer-Human Interaction*, 1(2):126–160, 1994.

[14] A. Martin and M. Ward. High dimensional brushing for interactive exploration of multivariate data. *Proc. IEEE Visualization*, pages 271–278, 1995.

[15] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. The QBIC project: querying images by content using color, texture and shape. *Proc. of SPIE Storage and Retrieval for Image and Video Databases*, pages 173–187, 1993.

[16] W. Peng, M. Ward, and E. Rundensteiner. Clutter reduction in multi-dimensional data visualization using dimension reordering. *Proc. IEEE Symposium on Information Visualization*, pages 89–96, 2004.

[17] Davood Rafiei and Stephen Curial. Effectively visualizing large networks through sampling. *Proc. IEEE Visualization*, pages 48–55, 2005.

[18] E. Riskin. Optimal bit allocation via the generalized BFOS algorithm. *IEEE Trans. on Information Theory*, IT-37:400–402, March 1991.

[19] R. Rosenholtz, Y. Li amd J. Mansfield, and Z. Jin. Feature congestion: a measure of display clutter. *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 761–770, 2005.

[20] D. Santa-Cruz, T. Ebrahimi, J. Askelof, M. Larsson, and C. Christopoulos. JPEG 2000 still image coding versus other standards. *SPIE's 45th annual meeting, Applications of Digital Image Processing XXIII*, 4115:446–454, Aug 2000.

[21] D. Scott. On optimal and data-based histograms. *Biometrika*, 66:605–610, 1979.

[22] S. Siggelkow. *Feature Histograms for Content-Based Image Retrieval*. Ph.D Thesis, University of Freiburg, Institute for Computer Science, Freiburg, Germany, 2002.

[23] M.J. Swain and D.H. Ballard. Color indexing. *International Journal of Computer Vision*, 7:11–32, 1991.

[24] P.B. Gibbons Swarup Acharya and Viswanath Poosala. Congressional samples for approximate answering of group-by queries. *Proc. ACM SIGMOD International Conference on Management of Data*, pages 487–498, 2000.

[25] S.K. Thompson. *Sampling*. John Wiley and Sons, Inc., New York, 2th edition, 1992.

[26] E. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, CT, 1982.

[27] J. Yang, M. Ward, and E. Rundensteiner. Hierarchical exploration of large multivariate data sets. *Data Visualization: The State of the Art 2003*, pages 201–212, 2003.