Sandia National Laboratories: Proposal for a Clustering Tool

Team: Avani Gadani, Daniel Lowd, Brian Roney, Eric Wu
Harvey Mudd College

Dr. Belinda Thom (Advisor)
Kevin Boyack (Liaison)

October 17, 2002
Contents

1 Introduction ................................................. 2

2 Background .................................................. 2
   2.1 Terminology ............................................. 3
   2.2 Problem Definition ..................................... 4

3 Approach ..................................................... 4

4 Schedule ..................................................... 6
   4.1 Team Member Contributions ............................. 7

5 Tool Development Approach ................................ 7
   5.1 Development Environment ............................... 7
   5.2 Interface Description .................................... 7
   5.3 GUI Example ............................................. 8

6 Deliverables ................................................ 9

7 Bibliography ............................................... 10
1 Introduction

Clustering is an automated data analysis technique that partitions a data set into clusters, sets of similar data points. This is particularly useful in various informatics fields such as genomics and cross-citation analysis, where the amount of information is intractable to manual inspection. User supervision is still important for understanding and analyzing the resulting clusters. Since there are many possible clusterings for any given set of data, we plan to implement a tool that will both cluster data and allow side by side analysis of different clusterings. At a minimum, we will implement the single-link and k-means clustering algorithms, with the option of adding more algorithms as the need arises. The tool will be written in MATLAB for its rapid prototyping abilities. It will work on data sets of at least 500 to 10,000 two-dimensional data points, creating a clustering of approximately 2 to 80 clusters in no more than half an hour. We will attempt to cluster data more quickly in order to support interactive data exploration.

The two proposed clustering algorithms were chosen for their simplicity, speed, and ability to handle varying types of data sets. K-means clustering is a partitional algorithm which runs quickly but tends to result in clusters with convex shapes [1]. Single-link clustering, on the other hand, is a hierarchical algorithm that can deal with arbitrary shapes, potentially at the expense of simple clusters [1]. The various problems with different clustering algorithms mean that a cluster validity metric would be useful for judging the quality of a given clustering. We also intend to research and implement measures of cluster validity.

2 Background

The goal of clustering is to partition the elements of a dataset into clusters, or subsets, so that the elements of any given cluster are more similar to each other (by some metric) than to the elements of any other cluster. One of the challenges of clustering is that there is rarely an obvious “correct” clustering. Consider the set of points in Fig 1. There are several different ways one could group the points to obtain a reasonable clustering. In Fig 2, the set of points forming an ellipse are considered as one cluster, and the remaining groups form different clusters. In Fig 3, the ellipse is broken apart and the pieces are grouped with the outer clusters.

Figure 1: A two-dimensional set of data points to be clustered.
Figure 2: One possible clustering of the data. The ellipse is considered as one group, since it forms a simple, connected shape.
Figure 3: Another possible clustering of the data. Here, the ellipse is broken apart and grouped with the outer clusters.

As this example showed, there are many ways to cluster a data set. Correspondingly, there are many clustering algorithms of varying quality. Currently, the choice of which algorithm to use often relies more on personal preference than an actual analysis. We would like to research metrics to help evaluate clustering algorithms more rigorously.
To do so, we will begin with some of the existing cluster validity tests. We propose to use Monte Carlo methods as one measure of the validity of a clustering. We also plan to research fuzzy cluster analysis and standard distance and statistical metrics, such as $\chi^2$ analysis.

In order to be valid, a clustering should define an unusual partition of a data set such that the partition is in some way specifically applicable to the data set. One way of testing this is the use of the null hypothesis. The null hypothesis $H_0$ assumes that in a given data set, there is in fact no particular structure at all, so any clustering is equally valid. Variants exist for different types of clustering algorithms. The validity of a given clustering is supported if the null hypothesis can be disproved, i.e., the data set is not random and the given clustering is appropriate.

A necessary tool in determining the randomness of a data set is an index that indicates the existence of a structure within the data set. One such measure is Hubert’s $\Gamma$ statistic [1]. Hubert’s $\Gamma$ statistic works on two $n \times n$ proximity matrices on the same $n$ objects, where neither matrix has any implied relationship with the other. The $\Gamma$ statistic is stated formally as:

$$
\Gamma = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X(i,j)Y(i,j).
$$

More generally, it is the point-by-point correlation between the two matrices. High values of this statistic indicate a high similarity between the two matrices.

The procedure for accepting or rejecting $H_0$ then is as follows. First, we generate a random data set of the same size as the original data set and run our clustering algorithm on this random data set. Then, we apply whatever similarity indices we wish to use, e.g., Hubert’s $\Gamma$ statistic, on the both our clustering of real data and the clustering of random data. We empirically determine the expected value of the index for the random data and compare it to the value of the index on the original data. If the difference between these two indices is above a certain predetermined threshold, the we can reject $H_0$ because we have demonstrated the original data set is not random and that the partition is applicable to the original data set.

This research is relevant because Sandia works in various fields that require clustering of data. Researchers at Sandia desire a tool that would allow them to visually compare multiple clusterings and that would calculate various clustering metrics for each ordination.

### 2.1 Terminology

**cluster:** A grouping of data with some level of correlation. For our purposes, the correlation is represented by physical proximity on a scatter-plot of the data. (See Fig 3 for one example.)

**distance measure:** A function that determines the relative distance or dissimilarity of two members of a data set. For example, if the data set consisted of points in the x-y plane, Cartesian and Manhattan distances would be two possible distance measures.

**features:** Data present for each member of the data set, which can be used to compute the similarity or dissimilarity of two members.

**index:** A unique numerical identifier for a point in the original data set.

**ordination:** An assignment of points to coordinates on the x-y plane, based on a similarity measure and a stochastic algorithm used by vGOrd.
.sim file: A tab-delimited ASCII file with one row per pair of points and three columns: the first point name or identifier, the second point name or identifier, and their similarity. This file is generally the result of running a similarity measure on some of the features of the data set.

similarity measure: A function that determines the relative similarity of two members of a data set. For example, if the data set consisted of strings, the length of the largest common substring would be one possible similarity measure.

vxOrd: A program developed at Sandia National Laboratories that reads a .sim file and outputs a .coord file. It attempts to place similar points near each other and dissimilar points further from each other by using a stochastic method to minimize a global error function.

vxInsight: A program developed at Sandia National Laboratories that enables the visualization of ordinations.

2.2 Problem Definition

We ultimately want to be able to view scatter plots of different data sets, apply a clustering algorithm on the data and allow a user to interact with the given clustering in order to see how it applies to scatterplots generated using other algorithms. Additionally, we wish to provide a variety of clustering metrics to evaluate the correctness of the grouping.

3 Approach

Sandia’s data handling algorithm currently takes the input and reduces it to a two-dimensional representation using a program called vxOrd. This representation is then sent to Sandia’s viewer, vxInsight.

The main goal of our tool is:

- To allow the user to visually compare different coordinate files with features or random seeds.
- Provide functionality to cluster the data using several algorithms, and to automatically compare clusterings using known validation tools.
- To ultimately allow the user and the clustering algorithm to work together to produce a better clustering solution than either could individually.

It will be a MATLAB-based tool for Microsoft Windows that will have file-level interaction with vxOrd, and possibly vxInsight. The tool will allow users to view scatter plots of data sets under different clustering algorithms or different similarity metrics from vxOrd. Users will also be able to select a set of points in one plot and see their corresponding locations in another plot.

K-means clustering is a commonly used partitional algorithm. The standard k-means clustering requires choosing a number of clusters (k) and assigns each data point a label corresponding with its cluster membership. The algorithm works as follows: First, randomly choose k cluster centers. Determine each data point’s cluster membership according to the cluster center closest to it by a
Figure 4: Sandia’s current process. Starting with a database of identifiers and information about them (features), a pairwise similarity file (.sim file) is generated using some similarity computation algorithm operating on a subset of the features. Using vxOrd, each identifier is then mapped to an x-y coordinate so that the Cartesian distance between two points roughly reflects their similarity. Finally, vxInsight is used to visualize the data.

Figure 5: Extension to process, for comparing different similarity metrics. The tool we propose to develop will read multiple ordinations generated from different similarity files, which in turn were generated using different similarity measures.
distance, \( d \). Recompute the cluster centers based on the actual cluster membership. Then repeat until some criteria is met, such as the number of data points being reassigned to new clusters falling under a given threshold. The underlying k-means process minimizes

\[
E = \sum_{i=1}^{c} \sum_{x \in C_i} d(x, m_i)
\]

where \( m_i \) is the center of cluster \( C_i \).

The run-time for the k-means algorithm is \( O(k \circ n) \) for each iteration. However, the number of iterations necessary is unknown since standard k-means is not guaranteed to converge. In addition, clusterings produced by k-means are dependent on the starting points of the clusters. Variants of k-means exist which are guaranteed to produce an optimal clustering from any starting points [2]. The advantages of this algorithm are that it runs in time linear to the cardinality of the data set, and modifications to the algorithm can be made to guarantee an optimal partitioning [2]. An inherent issue in k-means clustering is that convex clusters tend to be produced when Euclidean distance is used as the distance measure.

Single-link clustering can broaden one’s search to include non-convex clusters, though the algorithm introduces other issues. Single-link is an agglomerative hierarchical algorithm, which assigns each data point to an individual cluster, then merges clusters together according to the minimum of all pairwise distances between clusters members. The advantage of merging clusters based on distances between distinct members of each cluster is that non-convex clusters may be formed, such as clusters consisting of concentric circles. An example of non-convex clusters can be seen in Fig 2.

However, this tendency may also produce clusters that are “chained”, or non-compact. In addition, the process of determining the minimum of distances is relatively computational expensive; the algorithm runs in \( O(n^2 \log n) \). This run-time is within our bounds, however, since with assuming a 1GHz processor,

\[
\frac{(10,000^2 \log 10,000)}{1,000,000,000 \text{ instructions/sec}} \div 60 \text{ sec/minute} \approx 16.7 \text{ minutes}
\]

4 Schedule

Our current plan is to meet the following deadlines:

Oct. 17th Deadline for preliminary basic cluster-viewing tool

Oct. 18th Site Visit, Sandia National Laboratories: Albuquerque, NM

Oct. 29th Presentation: Initial Peer Review of Project

Nov. 05th Finish basic cluster-viewing tool

Nov. 17th Deadline for report on existing clustering metrics

Nov. 17th Implement at least one clustering algorithm.
Nov. 24th  Decide on and implement at least one metric for one clustering algorithm. Implement 2nd clustering algorithm if we have time.

Dec. 5th  Draft of Mid-year report
Dec. 10th  Presentation
Dec. 12th  Mid-year report due
Apr. 18th  Draft of Poster Due
Apr. 22nd  Code Freeze
Apr. 25nd  Poster Due
Apr. 29nd  Final Presentation Practice
May 1st  Draft Final Report/CD
May 9th  Final Report/CD

4.1 Team Member Contributions
Daniel and Brian are currently focused on implementing the tool in MATLAB, while Avani and Eric are researching the algorithms and validity metrics to be used.

5 Tool Development Approach
5.1 Development Environment
To quickly develop an extensible solution, we will use MATLAB as our programming language and development platform. The main disadvantages of MATLAB are that it creates some restrictions on GUI development and may be somewhat slower than an implementation in C or C++. Other languages may provide greater opportunities for customization and optimization, but we believe MATLAB to be the best choice for development because of the large amount of functionality built-in to MATLAB. A solution that would take months in Visual C++ can be done in weeks in MATLAB.

5.2 Interface Description
Project Design Specifications: We shall implement a function in MATLAB that will generate a GUI. This GUI will include:
- A file browsing system to select data sets to examine, each of which will be placed in a separate plot window.
- The ability to zoom, scale, and rotate any of the plots individually.
- The ability to select a subset of a plot, which will highlight the equivalent points in the other plot windows.
Figure 6: Freehand drawing of the proposed user interface design.

- Implementation of various clustering algorithms to cluster the data in a plot.
- Once clustered, display the input data as groupings. This permits the visual comparison of different clusterings.

Other features under consideration:

- Implementation of various clustering validity metrics, to examine the correctness of the different clustering runs.
- Implementation of additional clustering algorithms, or versions of the vxOrd/vxInsight algorithms that run in MATLAB.
- The ability to change a selected group of points to a different cluster in the same data set, to fix "common sense" mistakes that the user can see that the clustering algorithm cannot.
- The ability to select a plot and clustering algorithm, and having the clusters in that plot be displayed (by highlighting or coloring) in the additional plots.

5.3 GUI Example

As can be seen in the Fig 6, the proposed solution includes three main areas of view. A control section in the upper left consists of buttons to open a file, cluster the data, highlight points between plots, compare two data sets, or any other functionality that is decided on.

Below this is a sample menu for the "Cluster" button, allowing the user to select which algorithm to use. In this proposed solution, only the main plot is clustered, and the other plots have their
corresponding points labeled the same as the main plot. This will allow a quicker clustering, and will let the user see a quick contrast between the data sets.

To the right of the sample menu, the plots of data are displayed. The circles, dots, and X’s represent different colors or shapes to differentiate between clusters. In each plot there is an indication of the name of that data set, whether it is the main plot or not, and various commands, including the ability to select that window as the main plot, close the window, or rotate/zoom that plot independently of the others.

Paper clip included for humor value.

Figure 7 is intended to be another look at the same data. If you took the set of plot data in the first figure, closed the fourth window, selected the third plot as the main plot, and clicked on the “open” button, this is what should result. Note that the third plot only has two clusters, as compared to the initial example. Also, some data points are not included in the third data set, so they are indicated by a separate color (represented by squares) in the other plots.

6 Deliverables

- Create a Windows application using MATLAB that reads in data from given ordinations outputted by vxOrd, a software program provided by Sandia, and returns an x,y scatter graph of each of the inputs that is simply a dimensionality reduction from the original input.

It should allow the user to define a clustering by hand or run a variety of automatic clustering algorithms. If the user does not wish to visually validate each cluster, the program should calculate and display clustering metrics for each clustered plot. Both the number of clustering
algorithms and metrics will be extensible.
The user should be able to see several graphs side by side, and the application will also let
the user select and mark portions of one graph and have the same data points automatically
highlighted in the other graphs. The user will also be able to rotate and zoom an individual
graph.

- Investigate clustering algorithms and validity metrics to use with the application, or create our
own metric for the given data under a few well known clustering algorithms, such as k-means.

7 Bibliography

References
