Hierarchical Density Shaving: A clustering and visualization framework for large biological datasets

Gunjan Gupta Alexander Liu Joydeep Ghosh

Department of Electrical & Computer Engineering
University of Texas at Austin, Austin, TX 78712, USA.
{ggupta/aliu/ghosh}@ece.utexas.edu

Abstract

In many clustering applications for bioinformatics, only part of the data clusters into one or more groups while the rest needs to be pruned. For such situations, we present Hierarchical Density Shaving (HDS), a framework that consists of a fast, hierarchical, density-based clustering algorithm. Our framework also provides a simple yet powerful 2-D visualization of the hierarchy of clusters that can be very useful for further exploration. We present results to show the effectiveness of our methods.

1 Introduction

In many real-world clustering problems, only a subset of the data actually needs to be clustered. In particular, this is true of many types of large, high-dimensional bioinformatics datasets such as protein mass spectroscopy, phylogenetic profile data, and gene-expression datasets. In particular, gene-expression datasets measure expression levels of genes compared to a control across a few thousand genes over a set of experiments that cover only a specific "theme" such as stress-response, and therefore only a small number of genes related to the conditions show good clustering. From this data, biologists are interested in recovering clusters formed from small subsets of genes that show strongly correlated expression patterns, and that often map to the biological processes involved in the specific context (e.g., stress).

Density-based clustering algorithms [2, 1] use *kernel density estimation* [7] to identify dense regions, and are capable of clustering only a subset of dense data. Perhaps the first algorithm to use such an approach was *Hierarchical Mode Analysis*(HMA) [11], which could also find a compact hierarchy of dense clusters. However, HMA seems to have gotten lost in time (it was published in 1968) and is not known to most current researchers. In this paper, we present an improved framework called Hierarchy Density Shaving(HDS) that builds upon the HMA algorithm and greatly broadens its scope and effectiveness. These improvements include: (1) creating a faster version of HMA appropriate for larger datasets; (2) the ability to use a variety of distance metrics; and (3) a novel, useful visualization of the resulting cluster hierarchy that can help guide cluster selection. For more about the relationship between DBSCAN, HMA, and our algorithms, see [5]. Empirical tests of our HDS framework on gene expression microarray data show that we do indeed obtain very good results.

2 Hierarchical Mode Analysis

Let $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n \subseteq \mathbb{R}^d$ be a set of data points that need to be clustered. Let $\mathbf{M}_S$ represent the corresponding $n \times n$ symmetric distance matrix. The algorithms described in this paper only require $\mathbf{M}_S$ as input. In our algorithms, this distance metric could be Euclidean, 1 - Cosine Similarity, or Pearson Distance (d_p) [4] computed as $1 - p$, where p is the *Pearson Correlation*, a popular similarity measure for clustering gene-expression and other biological data [9, 8]. Justification for why these distance metrics can be used can be found in [5].

HMA [11] uses the following notion of density. Given some $r_\epsilon \in \mathbb{R} : min(\mathbf{M}_S) \leq r_\epsilon \leq max(\mathbf{M}_S)$ as an input, the density $\rho_{r_\epsilon}(\mathbf{x})$ at any given point $\mathbf{x}$ is proportional to the number of points in $\mathcal{X}$ that are within r_ϵ of $\mathbf{x}$ [1]: $\rho_{r_\epsilon}(\mathbf{x}) \propto |\{\mathbf{y} \in \mathcal{X} : d_S(\mathbf{y}, \mathbf{x}) \leq r_\epsilon\}$ The HMA algorithm is as follows [2]:

1. Select the density threshold as integer $n_\epsilon < n$, compute the inter-point distance matrix $\mathbf{M}_S$ and the distances $\mathbf{d}^{n_\epsilon}$ from each point to its n_ϵ^{th} nearest point.

2. Order the distances $\mathbf{d}^{n_\epsilon}$ so that the smallest is first using the array $\mathbf{a}^{n_\epsilon}$ as an index. Thus $\mathbf{a}^{n_\epsilon}$ defines the order in which the data points become dense: point $\mathbf{a}^{n_\epsilon}(1)$ has the smallest n_ϵ^{th} distance $\mathbf{d}^{n_\epsilon}(1)$ and is first

[1]The set of points within r_ϵ distance of $\mathbf{x}$ includes $\mathbf{x}$.

[2]Since [11] is not easily available, what follows below is presented exactly as in [11] except with the substitution of notation used in this paper.

to become dense when $r_\epsilon = \mathbf{d}^{n_\epsilon}(1)$, point $\mathbf{a}^{n_\epsilon}(2)$ is second at $\mathbf{d}^{n_\epsilon}(2)$, and so on.

3. Select distance thresholds r_ϵ from successive $\mathbf{d}^{n_\epsilon}$ values, initializing a new dense point at each cycle. As the second and each subsequent dense point is introduced, the method tests the new point to determine one of three possible fusion phases: either (i) the new point does not lie within r_ϵ of another dense point, in which case it initializes a new cluster mode, (ii) the point lies within r_ϵ of dense points from one cluster only, and therefore the point is directly fused to that cluster, or (iii) the point falls in the saddle region, lying within r_ϵ of dense points from separate clusters, and the clusters concerned are fused.

4. Finally, a note must be kept of the nearest-neighbor distance r_{min} between dense points of different clusters. When r_ϵ exceeds r_{min}, the direct fusion of the two clusters separated by r_{min} is indicated.

3 Density Shaving (DS)

For the labeled points (i.e., the dense points) from the i^{th} iteration of HMA, it can be shown that two dense points $\mathbf{x}, \mathbf{y} \in \mathcal{G}$ (where $\mathcal{G}$ is the set of dense points), belong to the same dense cluster represented as $\mathcal{C}$ if $d(\mathbf{x}, \mathbf{y}) < r_\epsilon$. That is: $\forall \mathbf{x}, \mathbf{y} \in \mathcal{G} : d(\mathbf{x}, \mathbf{y}) < r_\epsilon \Rightarrow \mathbf{x}, \mathbf{y} \in \mathcal{C}$ Thus, for any two points $\mathbf{x}_1$ and $\mathbf{x}_m \in \mathcal{G}$, if there exists a chain of points $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{m-1}, \mathbf{x}_m \in \mathcal{G}$ such that $\{d_S(\mathbf{x}_i, \mathbf{x}_{i-1}) < r_\epsilon\}_{i=2}^{m}$, then $\mathbf{x}_1$ and $\mathbf{x}_m$ also belong to the same cluster in a given iteration of HMA.

This leads to an algorithm that can compute the cluster labels in the i^{th} iteration of HMA directly without the iterative process required in HMA. We call this algorithm *Density Shaving* (DS), and a pseudocode for DS is presented in (Algorithm 1). DS essentially takes two parameters as inputs: (1) f_{shave}, the fraction of least dense points to *shave* or exclude from consideration, and (2) n_ϵ, the number of points that must be within a distance r_ϵ of a given point x_i in order for x_i to be considered dense, where DS computes the corresponding r_ϵ automatically as $r_\epsilon = \mathbf{d}^{n_\epsilon}(i)$, where $i = \lceil n(1 - f_{shave}) \rceil$. The output of DS consists of k clusters labeled 1 to k formed by the set $\mathcal{G}$ of n_c densest points and a "don't care" set $\mathcal{O}$ containing the remaining points which are labeled 0.

4 Hierarchical DS (HDS)

In this Section we describe *Hierarchical Density Shaving* (HDS), in which we first use the DS algorithm to compute a subset of the HMA iterations, and then perform a relabeling of the DS clusterings to generate a noise-tolerant version of the HMA hierarchy. We start with the following observation:

Algorithm 1 DS

```
Input: Distance matrix $\mathbf{M}_S$, $n_\epsilon$, $f_{shave}$.
Output: Cluster labels $\{lab_i\}_{i=1}^{n}$ corresponding to the $n$ data
    points.
    Initialize: $\{lab_i\}_{i=1}^{n} = 0$
    $n_c = \lceil n(1 - f_{shave}) \rceil$
5:  $[\mathbf{M}_{rad}^{nbr}, \mathbf{M}_{idx}^{nbr}] = sortrows(\mathbf{M}_S)$
    $[\mathbf{radx}^{n_\epsilon}, \mathbf{idx}^{n_\epsilon}] = sort(\mathbf{M}_{rad}^{nbr}(\cdot, n_\epsilon))$
    $r_\epsilon = radx^{n_\epsilon}(n_c)$
    $\mathcal{G} = \{\mathbf{x}(idx^{n_\epsilon}(i))\}_{i=1}^{n_c}$
    /* Lines 12-20: For each point in $\mathcal{G}$, find other dense points
10: within $r_\epsilon$ distance of it and make sure they have the same
    labels, if not, relabel */
    for $i = 1$ to $n_c$ do
        $idxb = binSearch(\{\mathbf{M}_{rad}^{nbr}(idx^{n_\epsilon}(i), j)\}_{j=n_\epsilon}^{n})$
        $\mathcal{X}_{nbrs} = \mathbf{M}_{idx}^{nbr}(idx^{n_\epsilon}(i), l)_{l=1}^{idxb}$
15:     $\mathcal{X}_{dnbrs} = \mathcal{X}_{nbrs} \cap \mathcal{G}$
        $L_{dnbrs} = unique(lab(\mathcal{X}_{dnbrs}))/\{0\}$
        $\forall y \in \mathbf{lab}$ if $\exists y \in L_{dnbrs} : y = i$
        $\mathbf{lab}(indexOf(\mathcal{X}_{dnbrs})) = i$
    end for
20: Count clusters: $k = |unique(\mathbf{lab})|/\{0\}$
    Remap the non-zero labels in $\mathbf{lab}$ to the range 1 to $k$.
```

Proposition 4.1 *The cluster labels in each of the n iterations of the HMA hierarchy can be computed independently of one another.*

This proposition follows as a consequence of the DS algorithm that can compute the i^{th} iteration of HMA directly without using the iterative procedure originally proposed by [11]. Computing all the n levels of HMA using either the original algorithm (Section 2) or DS results in a time complexity of $O(n^3)$. Since the HMA iteration cluster labels are nested (which follows from HMA directly), and because of Proposition 4.1, any subset of DS executions corresponding to some subset of the n HMA iteration clusterings also forms a hierarchical clustering. An obvious way to produce such a hierarchy faster would be to compute a linearly sampled subset of the n HMA iterations using DS. However, we propose an alternate scheme and an algorithm as follows:

Algorithm C: We start with all the $n_c = n$ data points in the first iteration, and at each successive iteration: (1) "shave" (remove) a fraction r_{shave}, $0 < r_{shave} \leq 1$ of the least dense of the n_c points from the current set of dense points [3], (2) apply DS steps 9 through 21 (Algorithm 1) on these n_c points. We repeat these two steps until n_c drops below 1. As output, Algorithm C produces an $n \times n_{iter}$ label matrix $\mathbf{L}$, where n_{iter} is the number of iterations in Algorithm C, and $\mathbf{L}(i, j)$ is the cluster ID of the ith point in

[3]Note that the geometric shaving of previous iteration's n_c results in a real number, which is then rounded off to the nearest integer to give the value n_c. In practice, we keep the real valued n_c and only round it when using it as an input to DS. Also, if the rounded value of n_c results in the same integer value as the last iteration, then that iteration is skipped.

the jth iteration. The parameter r_{shave} can be viewed as the exponent of an *exponential shaving*, and we refer to it as the "shaving rate". It can be shown that Algorithm C results in a total number of iterations that is at most $\lceil -\frac{\log(n)}{\log(1-r_{shave})} \rceil$, and that its time complexity is only $O(n^2 \log(n))$. The exponential shaving also gives HDS another desirable property: its ability to find finer approximations of the smaller, denser clusters, which matches well with our goal of finding small, pure, dense clusters.

We also define the HDS level $f_{level}(n_c)$ as the number of shavings by fraction r_{shave} required to obtain n_c, and it can be computed from n_c as follows:

$$f_{level}(n_c) = \frac{\log(n_c) - \log(n)}{\log(1 - r_{shave})} \qquad (1)$$

More details on HDS, including an even faster version called *Recursive HDS* are describe in [5].

Extracting a smoothed HMA hierarchy: The hierarchy produced by HMA involves top-down "growing" clusters; the algorithm starts with the densest point and then repeatedly merges an additional point in each iteration, either (1) starting a new cluster, (2) merging points in an existing cluster, or (3) merging two existing clusters. Although Algorithm C produces a subset of the full HMA iterations (represented by the $n \times n_{iter}$ label matrix $\mathbf{L}$), the labels are not based on labels in previous iterations. Hence, there is no correspondence between the different HDS iterations. In order to produce labels in HDS that also correspond to the HMA labels on a hierarchical basis, a re-labeling of the cluster labels needs to be performed. We perform the relabeling on the labels generated by Algorithm C using a bottom-up approach. Such a relabeling can also be viewed as a "compaction" of the hierarchy generated by Algorithm C.

Additionally, we introduce the notion of a *particle*, a cluster that emerges from a parent cluster but is considered spurious due to its small size. Particles are ignored when compacting $\mathbf{L}$. We define *particle* as any cluster of size less than n_{part} points. Note that $n_{part} = 0$ results in an exact subset of the HMA hierarchy, a larger value of n_{part} has a smoothing effect on the process of compacting the hierarchy, while a larger n_ϵ has a smoothing effect on the original HDS clustering. The effect produced by $n_{part} > 0$ is similar to that of *runt pruning* used in [10].

The relabeling proceeds as follows: (1) use the $uniquerows$ operation to find the unique cluster IDs at iteration $j-1$. (2) Repeat the following for each of the clusters found in Step 1: (2.1) If all points belonging to a cluster in iteration $j-1$ are either: (a) clustered in the same cluster in iteration j, (b) are assigned to the don't care set $\mathcal{O}$, or (c) are assigned to a cluster that is a particle, then we assign the child cluster at level j the label of the parent cluster at level $j-1$. That is, one can view the cluster on level j as a continuation of the corresponding cluster on level $j-1$, barring those points which are now part of $\mathcal{O}$ or a particle. If the condition in (2.1) is not satisfied, then (2.2) a cluster has split into two or more child clusters. Each of these child clusters is assigned a new cluster ID. The relabeled label matrix $\mathbf{L}_{HDS}$ represents a smoothed HMA hierarchy, which we refer to as the *HDS hierarchy*.

5 Visualization with HDS

Each row of the $n \times n_{iter}$ HDS hierarchy matrix $\mathbf{L}_{HDS}$ represents the cluster label of each point in all the HDS iterations. We now sort the rows of $\mathbf{L}_{HDS}$ using a dictionary sort, such that we give higher precedence to labels with smaller column indices.

A simple yet powerful visualization of the HDS hierarchy can be achieved by plotting this sorted matrix and assigning separate colors to each distinct positive value in the matrix, and a background color to the values that are 0. Figure 1, (g) shows such a visualization for the 2-D Sim-2 data, while Figure 2 (a) shows the same for the 6,151 dimensional Gasch data. The visualization also shows the *height* of each cluster, which is defined as the difference between the levels (Equation 1) of the first and the last iterations that the cluster existed in HDS. The visualization provides a powerful, compact, informative hierarchy and a spatially relevant 2-D projection of a high-dimensional density distribution. While many visualization tools are built on top of clustering results, the HDS visualization directly corresponds to the clustering methodology. The visual feedback provided also allows the user to explore the clusters discovered in the hierarchy as well as to adjust the smoothing parameters n_{part} and n_ϵ. Typically, however, the results are quite stable over a range of parameter values.

6 Experimental Evaluation

6.1 Datasets

We tested our framework on one real and one artificial datasets. The artificial dataset, called Sim-2, consists of 1,298 points generated from five 2-D Gaussians of different variances (which roughly correspond to the clusters in Figure 1 (f)) and a uniform distribution. The Gasch dataset [3], a widely used benchmark for testing clustering algorithms on microarray data, consists of 6,151 genes of yeast *Saccharomyces cervisiae* responding to diverse environmental conditions over 173 microarray experiments. Each of the 173 experiments have a description associated with them which was used to categorize the experiments into 11 classes.

In order to compare our algorithms with various benchmark algorithms, we used Adjusted Rand Index (ARI) [6], a metric which returns 1 for a perfect agreement between clusters and class labels and close to 0 when the clustering is as bad as random assignments. However, most labeled

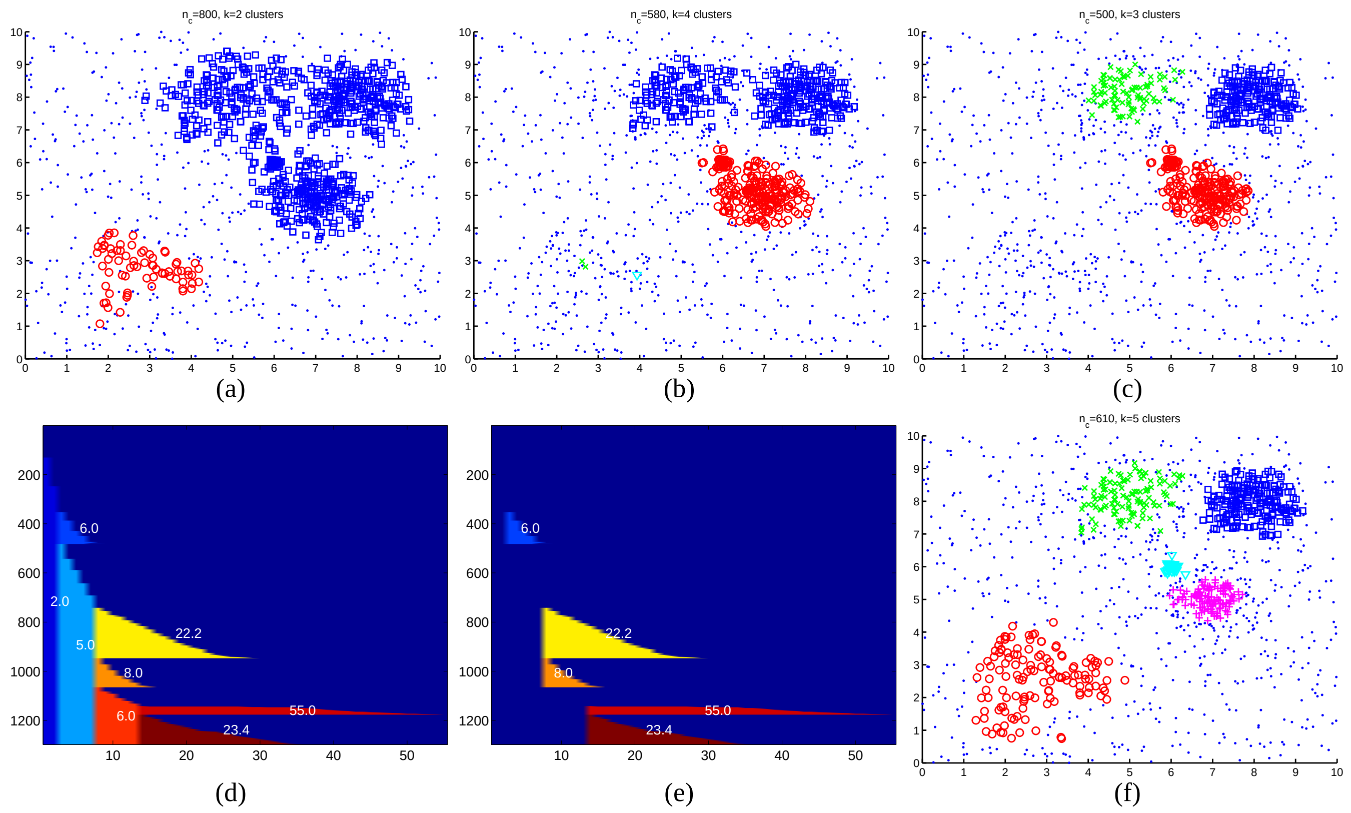


Figure 1. (a) to (c): Effect of DS applied with varying n_c (f_{shave}) for $n_\epsilon = 20$, resulting in a hierarchy of clusters. For $n_\epsilon = 20, n_{part} = 5$, HDS visualization after cluster identification (d), and manual cluster selection (e), allowing for clusters of different densities (f). The numbers on clusters in (d) and (e) represent the height of each of the clusters.

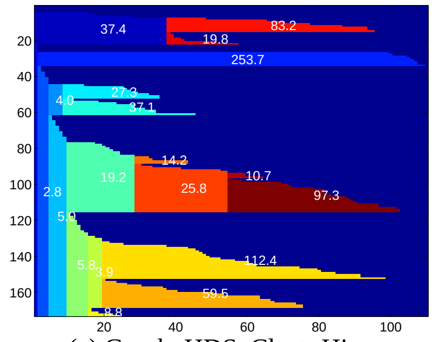
evaluation measures for clustering (including ARI) are sensitive to the number of clusters discovered and the percentage of data clustered. To get around this problem we ensure that the benchmark algorithms use the same n_c and k as our methods by applying the following procedure that we call *MaxBall*: (1) Find k clusters, where k is given by the number of clusters found by DS for a particular n_c. (2) Compute the cluster center for each of the k clusters as the mean of the cluster member points. (3) Assign each of the n points to their closest center and then pick n_c points closest to their cluster center. Assign remaining $n - n_c$ points to the "don't care" set. Using *MaxBall*, we modified K-Means and Single-Link agglomerative clustering.

For the sake of discussion, we define coverage as the fraction of points clustered (i.e., n_c/n). For DS, comparisons with other benchmarks were performed across a range of coverages. Since varying the coverage for DS results in varying k, the corresponding k is used as an input to the benchmark algorithms. We also include DBSCAN as a benchmark algorithm. Note that it is not possible to control either the number of clusters k or the coverage in DBSCAN. Instead, we fix $MinPts$ to 4 as suggested by [2] and then search for Eps that results in the desired coverage, while k is found automatically by DBSCAN.

6.2 Results

Figure 3, (top) and (bottom) compare DS with benchmarks on the Sim-2 and Gasch dataset using ARI over a range of fraction of data clustered (x-axis). In general, for lower coverages that correspond to dense regions, DS tends to perform very well. HDS works well for detecting the most significant dense regions in the data. The clusters also match well with the true labels in the target classes (e.g., Figure 2 (b) and (c), Figure 1, (f)). It should be stressed that since k is discovered by our framework and is given as an input to the MaxBall based benchmarks, they are not a viable alternative to our framework for finding dense regions automatically.

The hierarchy found by HDS on the extremely high-dimensional Gasch dataset is quite compact and easy to interpret (Figure 2(a)). Many of the clusters discovered by HDS (e.g., Figure 2(b)) contain highly correlated experimental descriptions, while others that form siblings have closely related descriptions. For example, a particularly



(a) Gasch, HDS, Clust. Hier.

Heat Shock 05 minutes hs-1 1
Heat Shock 10 minutes hs-1 1
Heat Shock 15 minutes hs-1 1
Heat Shock 20 minutes hs-1 1
Heat Shock 30 minutes hs-1 1
Heat Shock 40 minutes hs-1 1
Heat Shock 60 minutes hs-1 1
Heat Shock 80 minutes hs-1 1

(b) Gasch, Cluster $\mathcal{A}$, $|\mathcal{C}| = 8$

Heat Shock 17 to 37, 20 minutes 1
Heat Shock 21 to 37, 20 minutes 1
Heat Shock 25 to 37, 20 minutes 1
Heat Shock 29 to 37, 20 minutes 1
Heat Shock 33 to 37, 20 minutes 1
DBY7286 37degree heat - 20 min 12
DBYyap1-37degree heat-20 redo 9
DBY7286+0.3 mM H2O2 (20 min) 9
DBYyap1-+0.3 mM H2O2 (20 min) 9

(c) Gasch, Cluster $\mathcal{B}$, $|\mathcal{C}| = 9$

Figure 2. (a): Demonstration of HDS clustering and visualization of Gasch experiments showing the effectiveness of the 2-D projection of the 6,151 dimensional Gasch data; related clusters form “siblings" that are located close to each other; their size, density and heights are easy to compare. (b) and (c) show an example of such a sibling pair.

interesting pair of sibling clusters $\mathcal{A}$ and $\mathcal{B}$ are shown in Figure 2, (b) and (c). Both clusters contain heat shock experiments. However, the heat shock experiments in cluster $\mathcal{A}$ involve a constant heat (37 degrees) and variable time, while the heat shock experiments in cluster $\mathcal{B}$ involve variable heat and constant time. Additional examples can be found in an extended version of this paper [5].

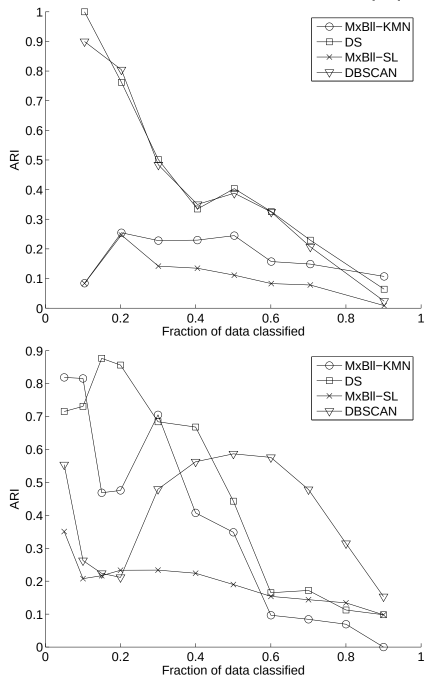


Figure 3. Comparison of DS with other methods using ARI on Gasch (left) and Sim-2 (right) datasets. Results for MaxBall K-Means were averaged over 10 (50) trials for the Gasch (Sim-2) dataset.

Acknowledgments: This research was supported by NSF grants IIS-0325116 and IIS-0307792. We would also like to thank anonymous reviewers of preliminary versions of this paper for providing useful comments.

References

[1] M. Ankerst, M. Breunig, H. P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure. In *ACM SIGMOD Conference*, 1999.

[2] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *In Proc. KDD-96*, 1996.

[3] Gasch A. P. et al. Genomic expression programs in the response of yeast cells to environmental changes. *Mol. Bio. of the Cell*, 11(3):4241–4257, December 2000.

[4] G. Gupta and J. Ghosh. Robust one-class clustering using hybrid global and local search. In *Proc. ICML 2005*, pages 273–280, Bonn, Germany, August 2005.

[5] G. Gupta, A. Liu, and J. Ghosh. Automatic hierarchical density shaving and gene diver. In *Technical Report, Dpt. of Elec. & Comp. Eng., Univ. of Texas at Austin, USA. Avail. at http://www.lans.ece.utexas.edu/∼gunjan/hds/hds-techreport.pdf*, 2006.

[6] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, pages 193–218, 1985.

[7] H. Jenq-Neng, L. Shyh-Rong, and A. Lippman. Nonparametric multivariate density estimation: a comparative study. *Science*, 42(10):2795–2810, October 1994.

[8] R. Mansson, P. Tsapogas, M. Akerlund, and et. al. Pearson correlation analysis of microarray data allows for the identification of genetic targets for early b-cell factor. *J. Biol. Chem.*, 279(17):17905–17913, April 2004.

[9] R. Sharan and R. Shamir. Click: A clustering algorithm with applications to gene expression analysis. *Proc. 8th ISMB*, pages 307–316, 2000.

[10] W. Stuetzle. Estimating the cluster tree of a density by analyzing the minimal spanning tree of a sample. *Journal of Classification*, 20:25–47, 2003.

[11] D. Wishart. Mode analysis: A generalization of nearest neighbour which reduces chaining effects. In *Proceedings of the Colloquium in Numerical Taxonomy*, pages 282–308, University of St. Andrews, Fife, Scotland, September 1968. Academic Press.