Subspace Clustering

Andrew Foss

PhD Candidate

Database Lab, Dept. of Computing Science
University of Alberta

For CMPUT 695 – March 2007

Motivation

- High Dimensional Issues
- Full Dimensional Clustering Issues
- Accuracy Issues

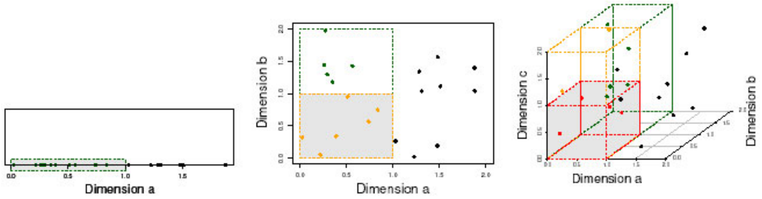
Curse of Dimensionality

- As dimensionality D $\rightarrow \infty$, all points tend to become outliers, e.g. [BGRS99]
- Clustering definition falters
- Thus, often little value in seeking either outliers or clusters in high D especially with methods that approximate interpoint distances

Exact Clustering

- Is expensive (how much?)
- Is meaningless since real world data is never exact
- Anyone want to argue for full D clustering in high D? Please do…

Increasing Sparcity



(a) 11 Objects in One Unit Bin (b) 6 Objects in One Unit Bin (c) 4 Objects in One Unit Bin

Full Space Clustering Issues

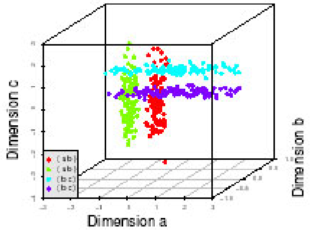
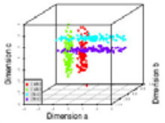
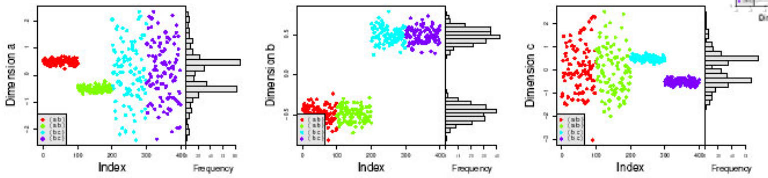


Figure 2: Sample dataset with four clusters, each in two dimensions with the third dimension being noise. Points from two clusters can be very close together, confusing many traditional clustering algorithms.

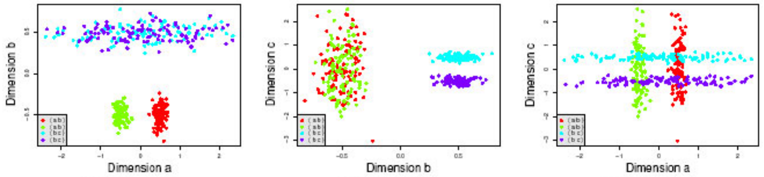
k-Means can't cluster this





(a) Dimension *a* (b) Dimension *b* (c) Dimension *c*

Figure 3: Sample data plotted in one dimension, with histogram. While some clustering can be seen, points from multiple clusters are grouped together in each of the three dimensions.



(a) Dims *a* & *b* (b) Dims *b* & *c* (c) Dims *a* & *c*

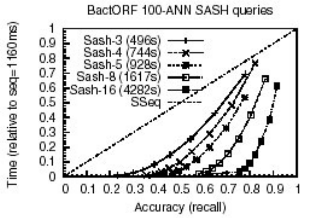
Figure 4: Sample data plotted in each set of two dimensions. In both (a) and (b) we can see that two clusters are properly separated, but the remaining two are mixed together. In (c) the four clusters are more visible, but still overlap each other are are impossible to completely separate.

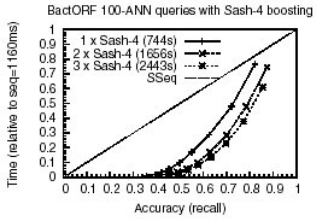
Approximation (Accuracy)

- D > 10, accurate clustering tends to sequential search

- Or inevitable loss of accuracy -

Houle and Sakuma (ICDE'05)





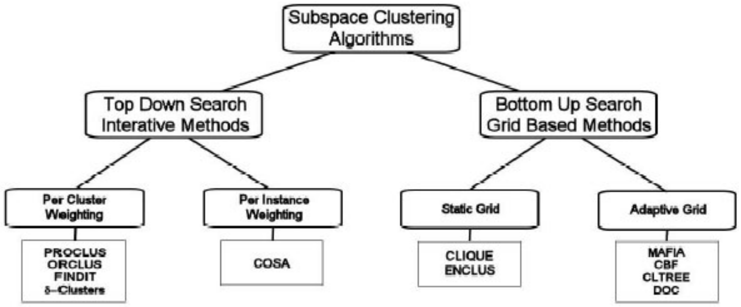
Why Subspace Clustering?

- Unlikely that clusters exist in the full dimensionality D
- Easy to miss clusters if doing full D clustering
- Full D clustering is very inefficient

Two Challenges

- Find Subspaces
  - Number exponential in D
- Perform Clustering
  - Efficiency issues still exist
- Can be done in either order

Approach Hierarchy [PHL04]


Subspace Clustering Algorithms
Top Down Search Interative Methods
Bottom Up Search Grid Based Methods
Per Cluster Weighting
Per Instance Weighting
Static Grid
Adaptive Grid
PROCLUS ORCLUS FINDIT δ–Clusters
COSA
CLIQUE ENCLUS
MAFIA CBF CLTREE DOC

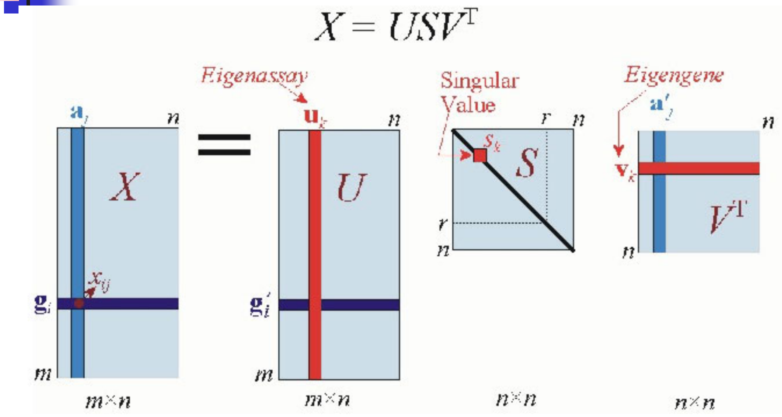
Three Approaches

- Feature Transformation + Clustering
  - SVD
  - PCA
  - Random Projection
- Feature Selection + Clustering
  - Search using heuristics to overcome intractability
- Subspace Discovery + Clustering

Feature Transformation

- Linear or even non-linear combinations of features to reduce the dimensionality
- Usually involves matrix arithmetic so expensive O(d^3)
- Global so can’t handle local variations
- Hard to interpret

SVD Example



http://public.lanl.gov/mewall/kluwer2002.html

SVD Example Output

Synthetic:
sine genes
(time series)
with noise
+ noise
genes

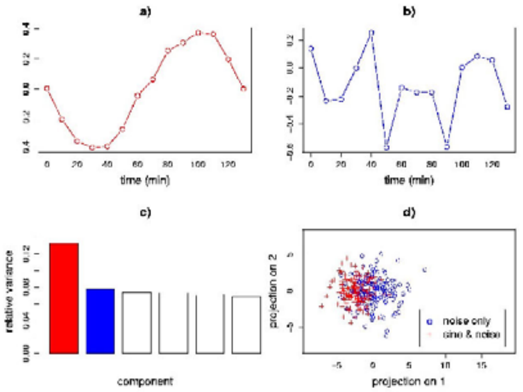


Figure 5.6. SVD-based detection of weak signals. a) A plot of the first eigengene shows the structure of the weak sine wave signal that contributes to the transcriptional response for half of the genes. b) The second eigengene resembles noise. c) A relative variance plot for the first six singular values shows an elbow after the first singular value. d) The signal and noise genes are not separated in an eigengene scatter plot of 150 of the signal genes, and 150 of the noise-only genes.

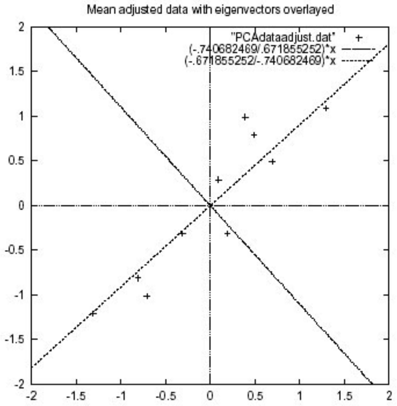
SVD Pros and Cons

- Can detect weak signals
- Preprocessing choices are critical
- Matrix operations are expensive
- If large singluar values *r* (< *n*) is not small, then difficult to interpret
- May not be able to infer action of individual genes

PCA

- Uses the covariance matrix, otherwise related to SVD
- PCA is an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on
- Useful only if variations in variance is important for the dataset
- Dropping dimensions may loose important structure – "…*it has been observed that the smaller components may be more discriminating among compositional group." – Bishop ' 05*

PCA Example



$$eigenvalues = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$eigenvectors = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

Figure 3.2: A plot of the normalised data (mean subtracted) with the eigenvectors of the covariance matrix overlayed on top.

http://csnet.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf

Covariance matrix

- Sensitive to noise. To be robust, outliers need to be removed but that is the goal in outlier detection
- Covariance is only meaningful when features are essentially linearly correlated. Then we don't need to do clustering.

Other FT Techniques

- Semi-definite Embedding and other non-linear techniques – non-linearity makes interpretation difficult.
- Random projections (difficult to interpret, highly unstable [FB03])
- Multidimensional Scaling – tries to fit into a smaller (given) subspace and assesses goodness [CC01]. Exponential number of subspaces to try, clusters may exist in many different subspaces in a single dataset while MDS is looking for one.

Feature Selection

- Top-down wrapper techniques that iterate a clustering algorithm adjusting feature weighting – at mercy of ability of full D clustering, currently poor due to cost and masking of clusters and outliers by sparcity in full D. E.g. PROCLUS [AWYPP99], ORCLUS [AY00], FindIt [WL02], δ-clusters [YWWY02], COSA [FM04]
- Bottom-up. Apriori idea, if a *d* dimensional space has dense clusters all its subspaces do. Bottom-up methods start with 1D, prune, expand to 2D, etc., e.g. CLIQUE, [AGGR98]
- Search: Search through subsets using some criterion, e.g. relevant features are those useful for prediction (AI)[BL97], correlated [PLLI01], or whether a space contains significant clustering. Various measures tried like ‘entropy’ [DCSL02] [DLY97] but not actually clustering the subspace (beyond 1D)

CLIQUE (bottom-up) [AGGR98]

- Scans the dataset building the dense units in each dimension
- Combines the projections building larger subspaces

CLIQUE Finds Dense Cells

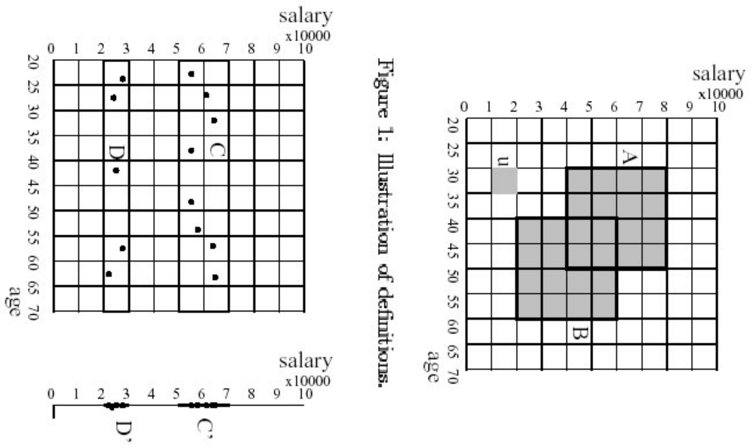


Figure 1: Illustration of definitions.

CLIQUE Builds Cover

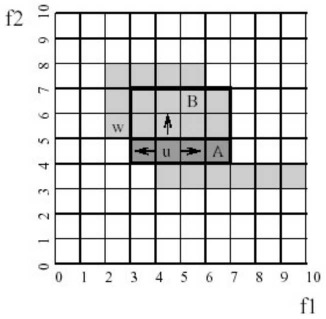


Figure 4: Illustration of the greedy growth algorithm.

CLIQUE

- Computes a minimal cover of overlapping dense projections and outputs DNF expressions
- Not actual clusters and cluster members
- Exhaustive search
- Uses a fixed grid – exponential blowup with D

CLIQUE Compared

100K synthetic data with 5 dense hyper-rectangles (dim = 5) and some noise

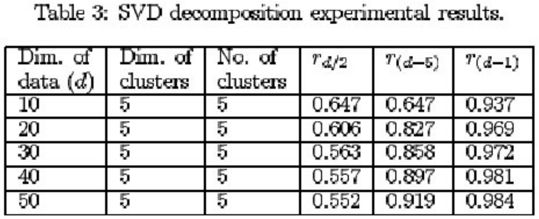


Table 3: SVD decomposition experimental results.

Dim. of data (d)	Dim. of clusters	No. of clusters	$r_{d/2}$	$r_{(d-5)}$	$r_{(d-1)}$
10	5	5	0.647	0.647	0.937
20	5	5	0.606	0.827	0.969
30	5	5	0.563	0.858	0.972
40	5	5	0.557	0.897	0.981
50	5	5	0.552	0.919	0.984

Only small difference between largest and smallest eigenvalues

CLIQUE Compared

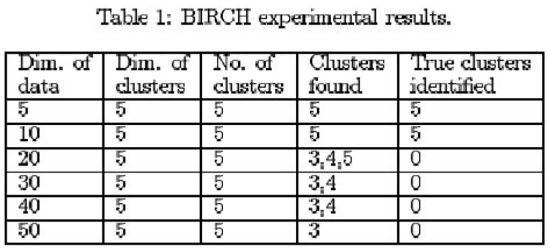


Table 1: BIRCH experimental results.

Dim. of data	Dim. of clusters	No. of clusters	Clusters found	True clusters identified
5	5	5	5	5
10	5	5	5	5
20	5	5	3,4,5	0
30	5	5	3,4	0
40	5	5	3,4	0
50	5	5	3	0

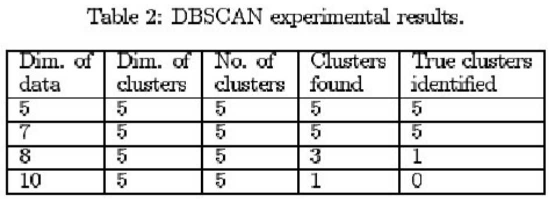


Table 2: DBSCAN experimental results.

Dim. of data	Dim. of clusters	No. of clusters	Clusters found	True clusters identified
5	5	5	5	5
7	5	5	5	5
8	5	5	3	1
10	5	5	1	0

Note: BIRCH - Hierarchical medoid approach, DBSCAN – density based

MAFIA [NGC01]

- Extension of clique that reduces the number of dense areas to project by combining dense neighbours (requires parameter)
- Can be executed in parallel
- Linear in N, exponential in subspace dimensions
- At least 3 parameters, sensitive to setting of these

PROCLUS (top-down) [AP99]

- *k*-Medoid approach. Requires input of parameters *k* clusters and *l* average attributes in projected clusters
- Samples medoids, iterates, rejecting 'bad' medoids (few points in cluster)
- First, tentative clustering in full D, then selecting *l* attributes on which the points are closest, then reassigning points to closest medoid using these dimensions (and Manhattan distances)

PROCLUS Issues

- Starts with full D clustering
- Clusters tend to be hyper-spherical
- Sampling medoids means clusters can be missed
- Sensitive on parameters which can be wrong
- Not all subspaces will likely have same average dimensionality

FINDIT [WL03]

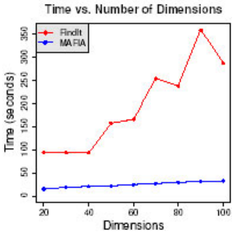
- Samples the data (uses subset S) and selects a set of medoids
- For each medoid, selects its V nearest neighbours (in S) using the number of attributes in which distance $d > \varepsilon$ (dimension-oriented distance *dod*)
- Other attributes in which points are close are used to determine subspace for cluster
- Hierarchical approach used to merge close clusters where *dod* below a threshold
- Small clusters are rejected or merged, various values of ε are tried and best taken

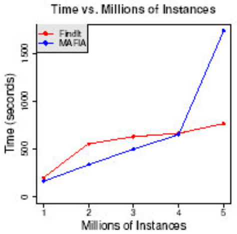
FINDIT Issues

- Sensitive to parameters
- Difficult to find low-dimensional clusters
- Can be slow because of repeated tries but sampling helps – speed vs quality

Parsons et al. Results [PHL04]

- MAFIA (Bottom-up) vs FINDIT (Top-down)





Cluster	1	2	3	4	5
Input	(4, 6, 12, 14, 17)	(1, 8, 9, 15, 18)	(1, 7, 9, 18, 20)	(1, 12, 15, 18, 19)	(5, 14, 16, 18, 19)
Output	(4, 6, 14, 17)	(1, 8, 9, 15, 18)	(7, 9, 18, 20)	(12, 15, 18, 19)	(5, 14, 18, 19)

Table 1: MAFIA misses one dimension in 4 out 5 clusters with $N = 100,000$ and $D = 20$.

Cluster	1	2	3	4	5
Input	(11, 16)	(9, 14, 16)	(8, 9, 16, 17)	(0, 7, 8, 10, 14, 16)	(8, 16)
Output	(11, 16)	(9, 14, 16)	(8, 9, 16, 17)	(0, 7, 8, 10, 14, 16)	(8, 16)

Table 2: FINDIT uncovers all of the clusters in the appropriate dimensions with $N = 100,000$ and $D = 20$.

Parsons et al. Results [PHL04]

- MAFIA (Bottom-up) vs FINDIT (Top-down)

Cluster	1	2	3	4	5
Input	(1, 5, 16, 20, 27, 58)	(1, 8, 46, 58)	(8, 17, 18, 37, 46, 58, 75)	(14, 17, 77)	(17, 26, 41, 77)
Output	(5, 16, 20, 27, 58, 81)	*None Found*	(8, 17, 18, 37, 46, 58, 75)	(17, 77)	(41)

Table 3: FINDIT misses many dimensions and and entire cluster at high dimensions with with $N = 100,000$ and $D = 100$.

Cluster	1	2	3	4	5
Input	(4, 6, 12, 14, 17)	(1, 8, 9, 15, 18)	(1, 7, 9, 18, 20)	(1, 12, 15, 18, 19)	(5, 14, 16, 18, 19)
Output	(4, 6, 14, 17)	(8, 9, 15, 18) (1, 8, 9, 18) (1, 8, 9, 15)	(7, 9, 18, 20)	(12, 15, 18, 19)	(5, 14, 18, 19)

Table 4: MAFIA misses one dimension in four out of five clusters. All of the dimensions are uncovered for cluster number two, but it is split into three smaller clusters. $N = 100,000$ and $D = 100$.

SSPC [YCN05]

- Uses an objective function based on the relevance scores of clusters – clusters with maximum number of relevant attributes is preferable. An attribute is relevant if the variance of its objects on a_i is low compared with D's variance on a_i (implication?)
- Uses a relevance threshold, chooses *k* seeds and relevant attributes. Objects assigned to cluster which gives best improvement
- Iterates rejecting 'bad' seeds
- Run repeatedly using different initial seed sets

SSPC Issues

- One of the best algorithms so far
- Sensitive to parameters
- Iterations take time but one may come out good
- Can find lower dimensional subspaces than many other approaches

FIRES [KK05]

- How to keep attribute complexity to quadratic?
- Builds a matrix of shared point count between 'base clusters'
- Attempts to build candidate clusters from *k* most similar

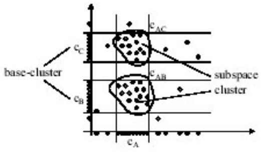


Figure 1. Overlapping subspace clusters

FIRES cont.

- Authors say 'Obviously [for cluster quality], cluster size should have less weight than dimensionality'. They use a quality function √(size).dim to prune clusters
- Do you agree?
- Alternatively, they suggest use of any clustering algorithm on the reduced space of base clusters and their points
- This worked better probably due to all the parameters and heuristics in their main method

EPCH [NFW05]

- Makes histograms in *d*-dimensional spaces by applying a fixed number of bins
- Inspects all possible subspaces up to size *max_no_cluster*
- Effectively projection clustering

EPCH

- Efficient only for *max_no_cluster* small

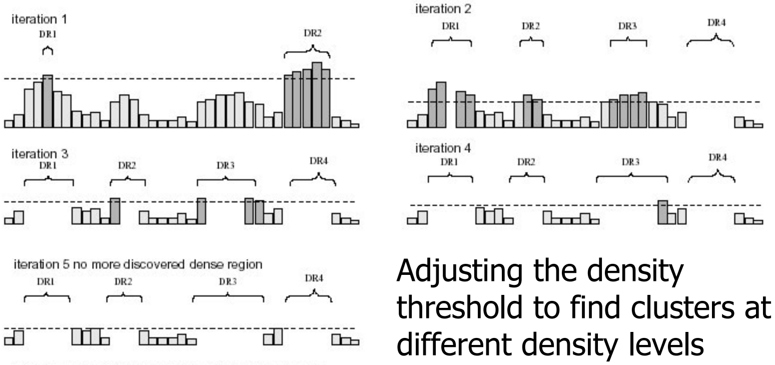


Fig. 4. Adaptive approach to iteratively lower the threshold value.

Adjusting the density threshold to find clusters at different density levels

DIC Dimension Induced Clustering [GH05]

- Uses ideas from fractals called intrinsic dimensionality
- Key idea is to assess local density around each point + density growth curve

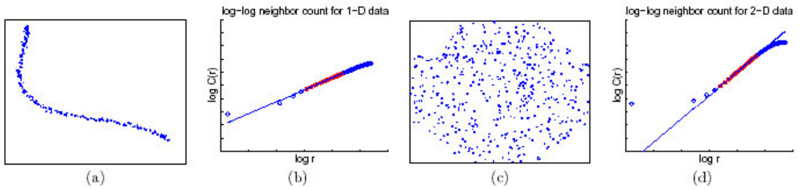


Figure 2: Intuition behind the intrinsic dimensionality (correlation dimension).

DIC

- Uses nearest neighbour algorithm (typically O(n^2))
- Each point x_i is characterised by its local density d_i and d_i‘s rate of change c_i
- These pairs are clustered using any clustering algorithm

DIC

- Claim: method independent of dimensionality but don't address sparcity issues, NN computation issues
- Two points in different locational clusters but with closely similar local density patterns can appear in the same cluster. Authors suggest separation using single-linkage clustering.
- Also suggest using PCA to find directions of interest. Otherwise can't find regular subspaces.
- Many similarities in core idea to TURN* but without resolution scan. DIC fixes just one resolution.

Conclusions

- Many approaches but all tend to run slowly
- Speedup methods tend to cause inaccuracy
- Parameter sensitivity
- Lack of fundamental theoretical work

References

- **[AGGR98]** R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data*, 1998.
- **[AP99]** C. Aggarwal, C. Procopiuc, JL Wolf, PS Yu and JS Park. Fast algoritjms for projected clustering. In *SIGMOD*, 1999.
- **[AY01]** C. Aggarwal and P. Yu. Outlier detection for high dimensional data. In *Proc.of ACM SIGMOD Conference*, pp. 37-46, 2001.
- **[BGRS99]** K Beyer, J Goldstein, R Ramakrishnan, and U Shaft. When is "nearest neighbour" meaningful? In *Proc. of the Intl. Conf. on Database Theory (ICDT 99)*, pp. 217–235, 1999.
- **[BL97]** A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, Vol. 97, pp. 245-271, 1997.
- **[CC01]** T. Cox and M. Cox. *Multidimensional Scaling*. Chapman Hall, 2nd edition edition, 2001.
- **[FB03]** Xiaoli Z. Fern and Carla E. Brodley. Random Projection for High Dimensional Data Clustering: A Cluster Ensemble Approach. *Proceedings of the Twentieth International Conference of Machine Learning*, 2003.
- **[GH05]** A Gionis, A Hinnenburg, S Papadimitriou and P Tsaparas. Dimension Indiced Clustering. In *KDD*, 2005.
- **[HXHD04]** Z. He, X. Xu, J.Z. Huang and S. Deng. A frequent pattern discovery based method for outlier detection. In *Proc. of WAIM'04*, pp. 726-732, 2004.
- **[HXD05]** Zengyou He, Xiaofei Xu, and Shengchun Deng. A Unified Subspace Outlier Ensemble Framework for Outlier Detection in High Dimensional Spaces. Posted May 2005. http://arxiv.org/abs/cs.DB/0505060
- **[KK05]** HP Kriegel, P Kroeger, M Renz and S Wurst. A generic framework for efficient subspace clustering ofhigh-dimensional data. In *ICDM*, 2005.
- **[MY97]** R. Miller and Y. Yang. Association rules over interval data. In *Proc. ACM SIGMOD International Conf. on Management of Data*, pages 452-461, 1997.
- **[NFW05]** EKK Ng, AWC Fu and RCW Wong. Projective clustering by histograms. *IEEE TKDE*, 17, pg. 369-383, 2005.
- **[NGC01]** H nagesh, S Goil and A Choudhary. Adaptive grids for clustering massive data sets. In *SDM*, 2001.
- **[PHL04]** L. Parsons, E. Hague and H. Liu, Subspace clustering for. high dimensional data: a review. *SIGKDD Explorations*,. Vol. 6 (1), pp. 90-105, 2004.
- **[PLLI01]** Pena, J. M., Lozano, J. A., Larranaga P., and Inza, I., Dimensionality reduction in unsupervised learning of conditional Gaussian networks, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23 (6), pp. 590-630, 2001.
- **[WS04]** K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidenite programming. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR-04)*, volume II, pages 988.995, 2004.
- **[WL03]** KG Woo, JH Lee, MH Kim and YJ Lee. FINDIT: a fast and intelligent subspace clustering algorithm using dimension voting. *Information andSoftware Technology*, 6, pg. 255-271, 2003.
- **[YCN05]** KY Yip, DW Cheung and MK Ng. On discovery of extremely low-dimesnional clusters using semi-supervised project clustering. In *ICDE*, 2005.