Hierarchical Modal Association Clustering

Surajit Ray and Yansong Cheng

November 29, 2010

1 Introduction to Modal Clustering

Model-based clustering has been widely used for clustering heterogeneous populations. But standard model based clsutering are often limited by the shape of the component densities. In this document, we describe a mode associated clustering approach (Li et al 2007) applying new optimization techniques to a nonparametric density estimator. A cluster is formed by those sample points that ascend to the same local maximum (mode) of the density function. The path from a point to its associated mode is efficiently solved by an EM-style algorithm, namely, the Modal EM (MEM). This clustering method shares the major advantages of mixture model based clustering. Moreover, it requires no model fitting and ensures that every cluster corresponds to a bump of the density. A hierarchical clustering algorithm is also developed by applying MEM recursively to kernel density estimators with increasing bandwidths. The kernel density estimate becomes smoother with the increase of smoothing parameters and more points tend to converge to the same mode. This suggests a hierachical clustering approach by choosing a serial of increasing smoothing parameters, namely Hierarchical Modal Association Clustering(HMAC).

Historically cluster analysis techniques has been approached from either a fully parametric view (the model based approach), e.g. mixture model based clustering, or a distribution free approach, e.g. the linkage based clustering. While the parametric paradigm provides the inferential framework and accounts for the sampling variability, it often lacks the flexibility to accommodate complex clusters and are not scalable to high dimensional data. On the other hand, the distribution free approaches are often fast and are capable of uncovering complex clusters by making use of different distance measures, but the inferential framework is distinctly missing. Modal clustering kneads the strengths of these two seemingly different approaches and to develop the new paradigm of nonparametric modal clustering which can accommodate flexible subpopulation structures and are computationally fast and scalable to high dimension, but retains the much desired natural inferential framework of parametric mixtures.

1.1 Parallel implementation of Modal Clustering

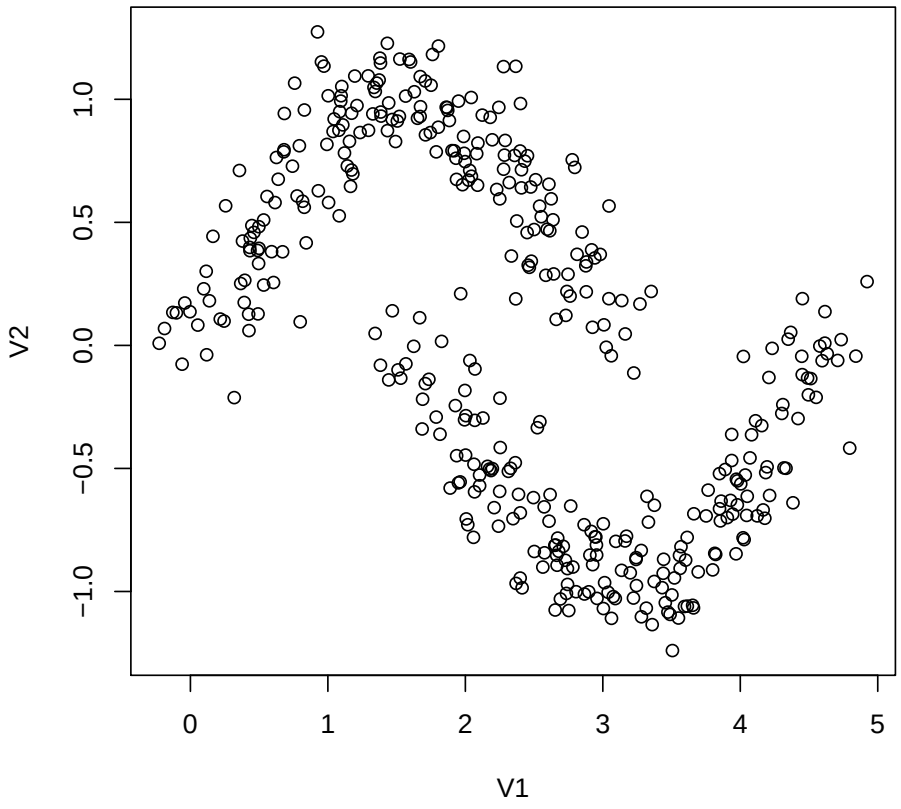
In this package we also develop a methodology for parallelization of modal clustering. We simply divide the data (number of samples) into the number of parallel nodes specified by the user. Ideally one would want to use all available processors of one's machine or nodes in a cluster. The first step is to construct modal clusters in parallel for each of these partitions and then merge the modes for higher bandwidths to provide a full hierarchical cluster. One should note that even if the user has one processor available, the user, partitioning large data into several partitions gives the advantage of "divide and conquer" by significantly reducing the computational burden. The parallel hierarchical modal association clustering (pHMAC) algorithm provides the main function `phmac` in this package.

2 Use of functions

In this section we demonstrate the use of various functions in this package by applying it on a few example datasets.

First, consider the data in a dataset which resembles broken parts of a disc as given in following figure. We simulated this dataset with non-gaussian cluster structures. In fact each cluster is embedded in a lower dimensional manifold (1-d curve).

```
> library(Modalclust)
> data(disc2d)
> plot(disc2d)
```



One can use Modal clustering to cluster this dataset in the following three ways.

(i) Standard Modal clustering, i.e. without partition

(ii) Partitioning the dataset into *npart* partitions but running them sequentially in a single processor.

(iii) Partitioning the dataset into *npart* partitions but running them in parallel using *npart* processors. This requires the use of package *multicore*

To cluster the `disc2d` using the non-partition mode (provided by default parameters in function `phmac`)

```
> disc2d.hmac = phmac(disc2d, npart = 1, parallel = F)

Performing initial Modal clustering..........
Building hierarchical Modal clusters at
```

```
level  1 ...level  2 ...level  3 ...level  4 ...level  5 ...
level  6 ...level  7 ...level  8 ...level  9 ...level  10 ...
```

On the other hand one can use the parallel mode to cluster the same data. To partition the dataset into 4 partitions and to run them sequentially in a single processor one can use

```
> disc2d.part=phmac(disc2d,npart=4,parallel=F)
```

To partition the dataset into 4 partitions and to run them in parallel in 4 processor one should use

```
> disc2d.phmac = phmac(disc2d, npart = 4, parallel = T)

Partioning Data
Using parallel computing for performing initial Modal clustering
....................................
Building membership from initial partitions
Building hierarchical Modal clusters at
level  1 ...level  2 ...level  3 ...level  4 ...level  5 ...
level  6 ...level  7 ...level  8 ...level  9 ...level  10 ...
```

Note that the output of the parallel and the non-parallel version will be slightly different, as the random partitions of the data may provide overlapping clusters in the first stage. We are in the process of developing methods to overcome this. But partitioning the data provides a huge numerical advantage and it might be the most practical solution for clustering large number of observations.

The function `phmac` returns an object of class "hmac". The summary of the output of such "hmac" objects can be obtained by invoking the command

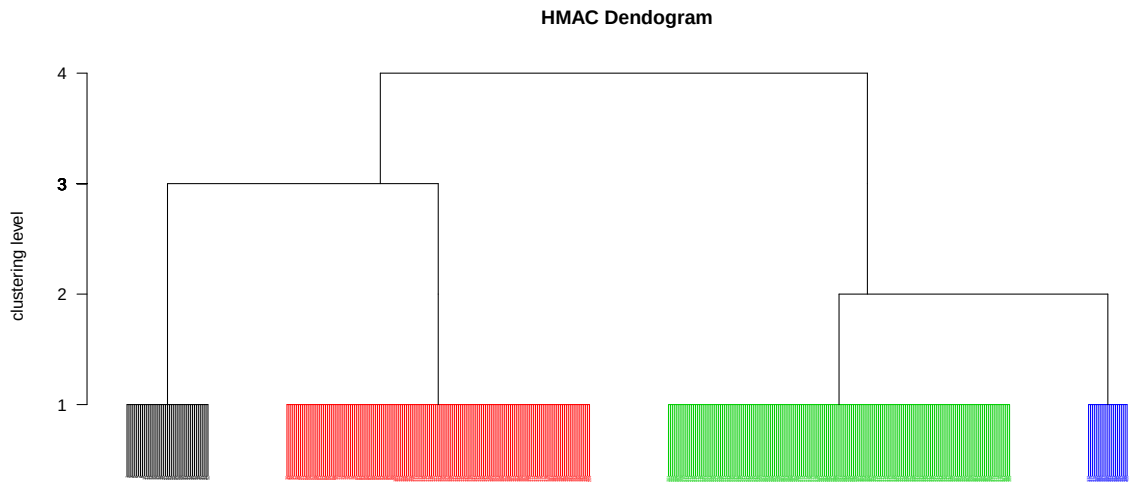
```
> summary(disc2d.phmac)

At levels 1 2 3  there are  4 3 2  clusters respectively

The modes of each level of clusters are

Modes at level  1
         X1         X2
1 0.4991144  0.4112434
2 1.4565542  0.9594440
3 2.9901931 -0.8986372
4 4.4221910 -0.1448845

Modes at level  2
         X1         X2
1 0.5184229  0.4269087
2 1.4780933  0.9519242
3 3.0214888 -0.8986338

Modes at level  3
        X1         X2
1 1.502569  0.9424770
2 3.049604 -0.8960544
```

2.1 Plotting methods and extracting clusters at specific level

There are several plotting function one can use to visualize the output of `phmac` function, i.e. an "hmac" object.

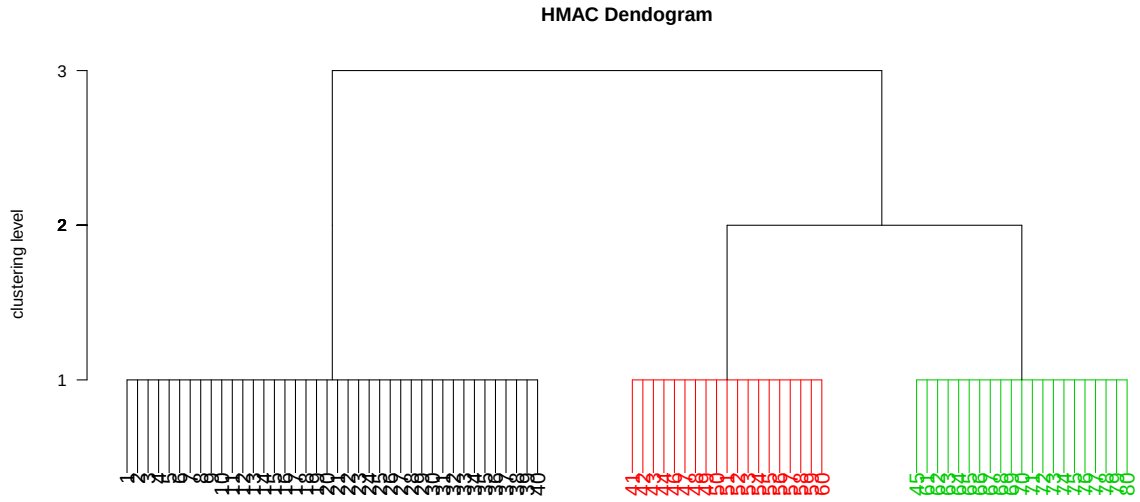
To visualize the full hierarchical tree one can use the following command.

```
> plot.hmac(disc2d.hmac)
```



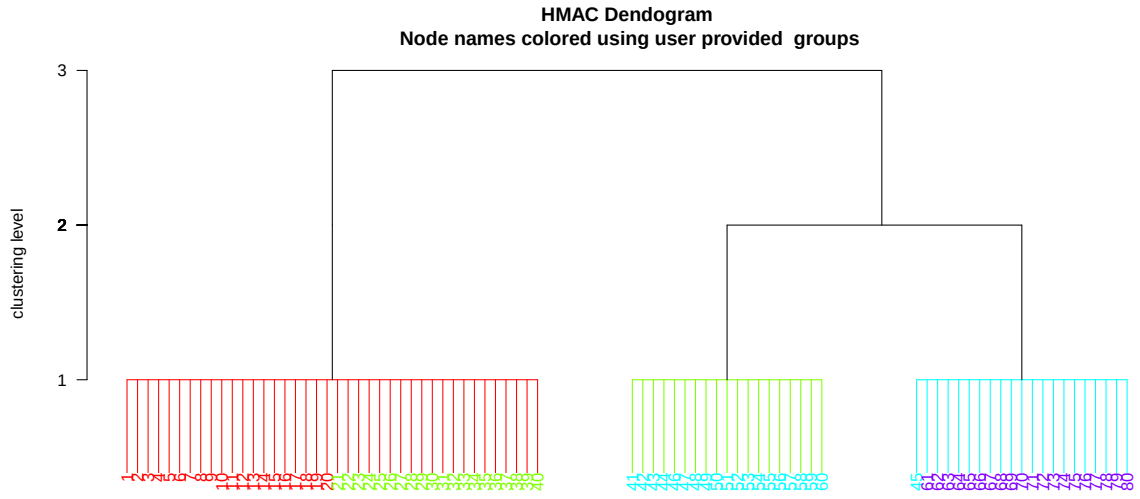
A clearer picture of the tree structure along with their original membership is visible in the following example with 80 data points having 2 big clusters and 4 sub clusters. This hierarchical layer is visible in the following graph. Note that one of the subclass was not separable as their modes merged the two clusters.

```
> set.seed(20)
> mix4 = data.frame(rbind(rmvnorm(20, rep(0, 4)), rmvnorm(20, rep(2,
+     4)), rmvnorm(20, rep(10, 4)), rmvnorm(20, rep(13, 4))))
> mix4.hmac = phmac(mix4, npart = 1)

Performing initial Modal clustering..........
Building hierarchical Modal clusters at
level  1 ...level  2 ...level  3 ...level  4 ...level  5 ...
level  6 ...level  7 ...level  8 ...level  9 ...level  10 ...

> plot(mix4.hmac)
```



For validation, One can also verify the output of the clustering and hence the tree with groups provided by user, using the `userclus` option, such as

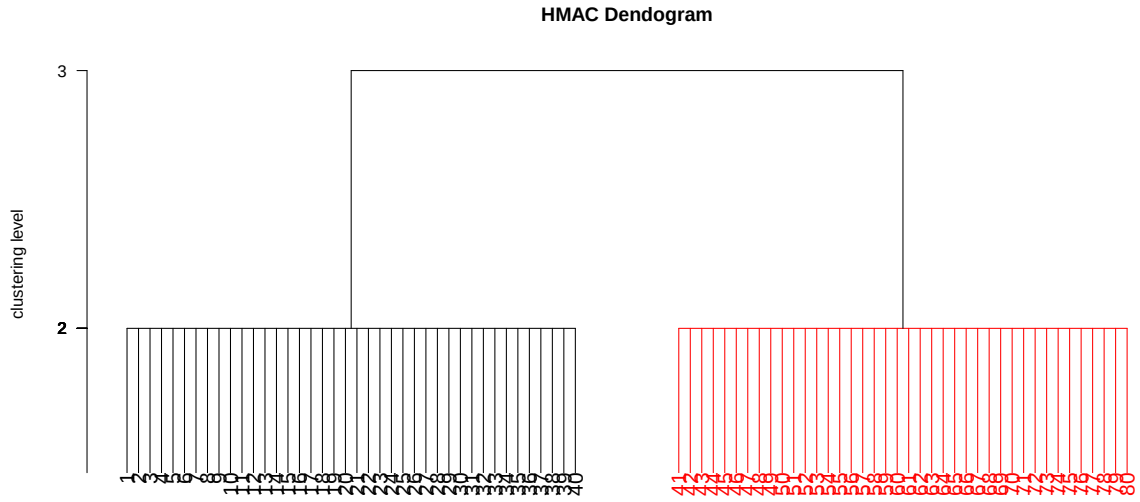
```
> plot(mix4.hmac, userclus = rep(c(1, 2, 3, 4), each = 20))
```



Moreover one can specify the number of clusters or the level from which one wishes to start the tree.

```
> plot(mix4.hmac, n.cluster = 2)
```

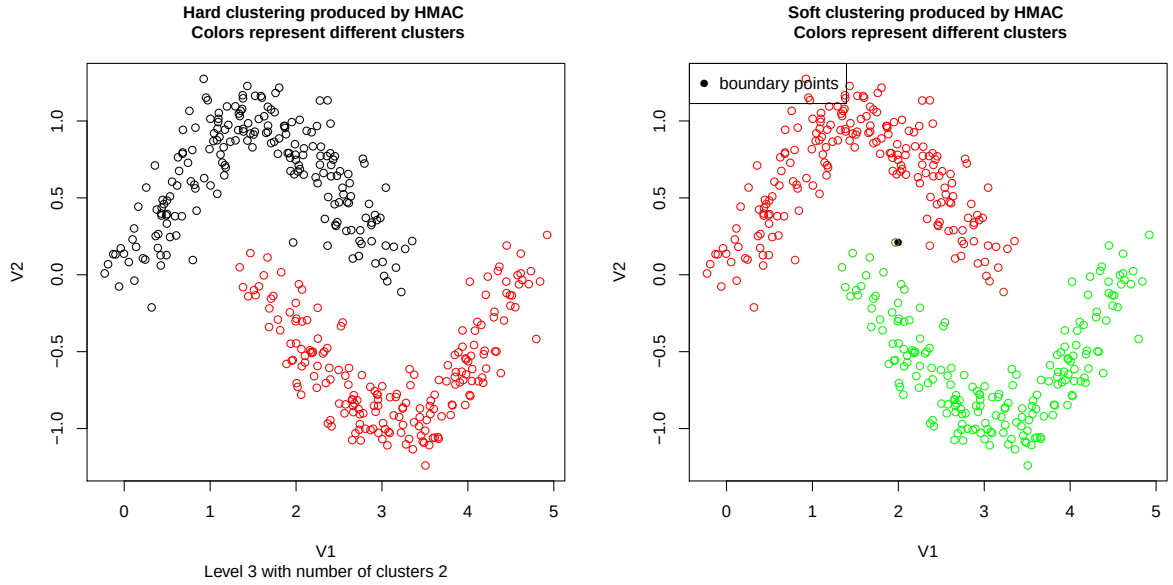
```
The level at which there are 2 clusters is 2
```



Alternatively one can use the `hard.hmac` or `soft.hmac` to visualize or extract the hard or soft clustering of a specified level or a user specified number of cluster.

To get the hard and soft cluster plot, either the number of clusters or the clustering level needs to be specified.

```
> par(mfrow = c(1, 2))
> hard.hmac(disc2d.hmac, n.cluster = 2)

The level at which there are 2 clusters is 3

> soft.hmac(disc2d.hmac, level = 3)
> par(mfrow = c(1, 1))
```

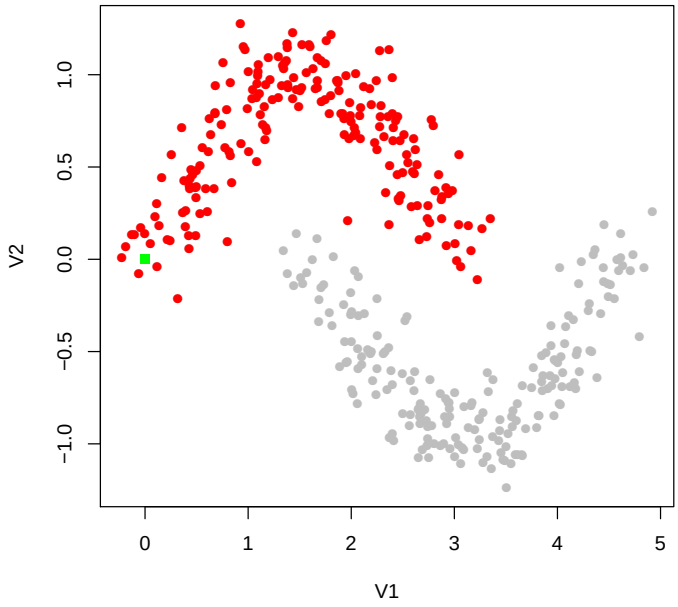


With the `plot=F` option `hard.hmac` will return the membership of each obsearvation while `soft.hmac` will return the posterior probability of each point and the boundary point as well.

```
> member = hard.hmac(disc2d.hmac, n.cluster = 2, plot = F)

The level at which there are 2 clusters is 3

> member

  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[112] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[149] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[186] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[223] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[260] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[297] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[334] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[371] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

> member.soft = soft.hmac(disc2d.hmac, level = 3, plot = F)
> head(member.soft$post.prob)

          [,1]         [,2]
[1,] 0.9999658 3.415439e-05
[2,] 0.9999233 7.670755e-05
[3,] 0.9997912 2.087752e-04
[4,] 0.9975377 2.462327e-03
[5,] 0.9996352 3.648461e-04
[6,] 0.9997089 2.911416e-04
```

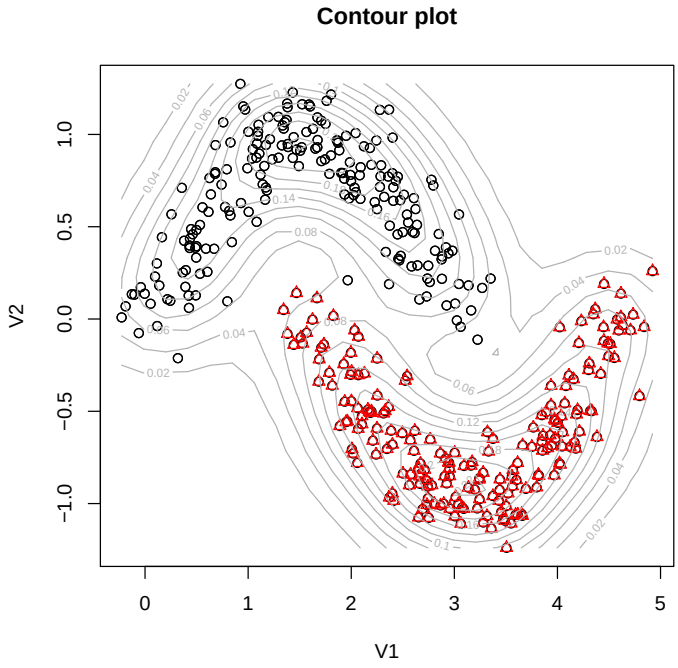
For the convenience of application, users can also identify whole clusters of point by either clicking the point on the graph or specifying the point as an input in the `choose.cluster` function. The clicked/specified point along with other points within the same cluster will be highlighted.

```
> choose.cluster(disc2d.hmac, n.cluster = 2, x = c(0, 0))

The level at which there are 2 clusters is 3
```



In addition in this package, one can also visualize the contour plot with specified number of clusters.

```
> contour.hmac(disc2d.hmac, n.cluster = 2, col = gray(0.7))

The level at which there are 2 clusters is 3
```



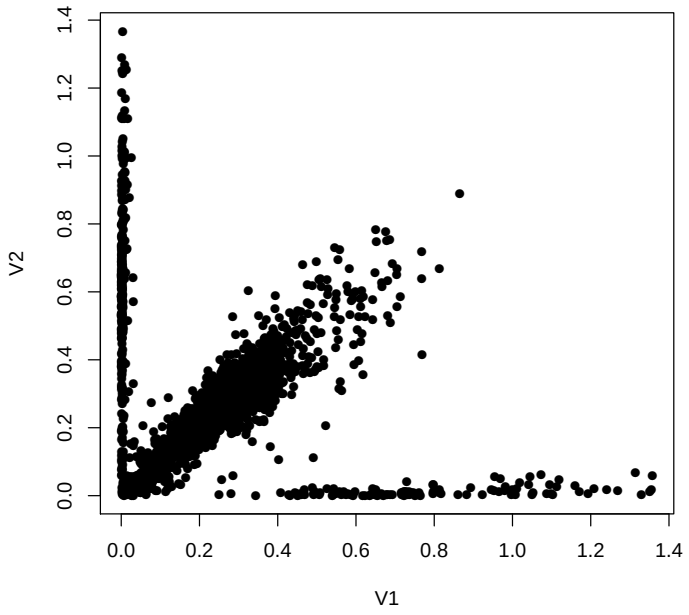
3 Further Examples

In the previous section we demonstrated the use of the functions in this package using a two dimensional data. Our method is neither restricted to two dimensions nor is it restricted to specific shapes of the clusters. Here we provide more examples, some on real data and some on simulated data to demonstrate other features of the functions available in this package.

First we focus on a two dimensional data obtained from high throughput genotyping from Illumina technology named **GOLDEN GATE**. [`http://www.illumina.com/technology/goldengate_genotyping_assay.ilmn`] The data values are actual measurements made by the machine (intensity), after these are normalized (background subtracted etc). The data set is used for making genotype calls by Illumina. The data around X- and Y-axes represents the two homozygous genotypes (e.g. AA and TT), while the cluster along the 45-degree line represents the heterozygous (e.g. AT) genotype. Due to noisy reads, the data points often lie in-between the axes, and cluster detection is used for making automatic genotype calls.

The scatter plot of the original data is shown below. The non-gaussian density shape and the abundance of points neear the axis plays an important role in clustering these points and Modal Clustering captures it very well.

```
> data(cta20)
> data(cta20.hmac)
> plot(cta20, pch = 16)
```

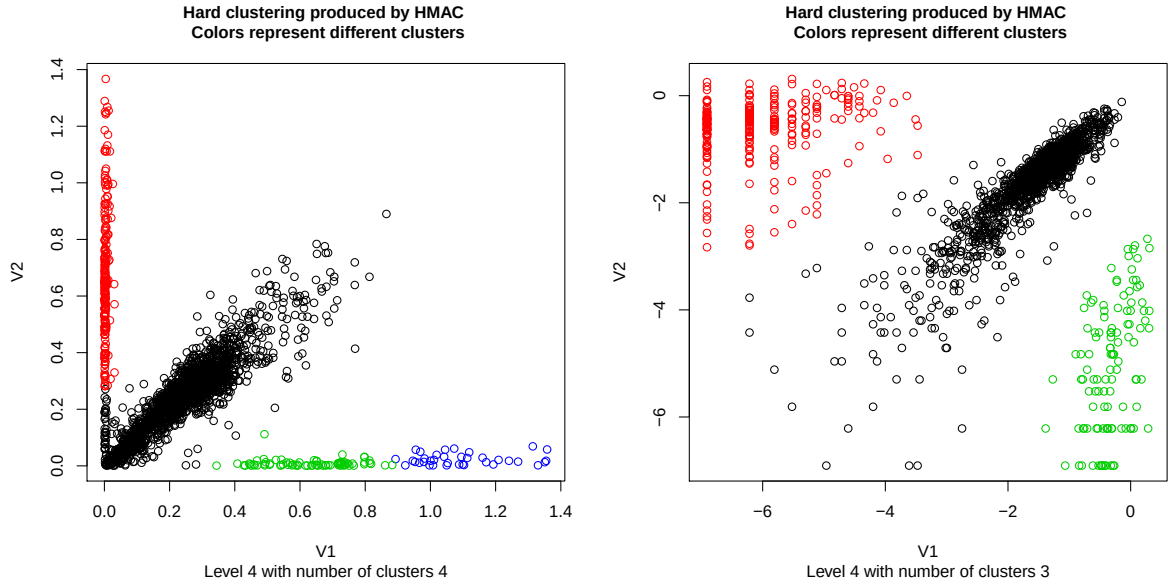


One can always apply a log transformation or some other more general transformations like Box Cox or logicle, but they are not designed to make clustered data normal. In fact even after log transformation the data are skewed and hence standard clustering algorithm failed to capture the true clustering whereas the Modal Clustering method could capture the three clusters in the transformed scale too.

Here we provide the clustering of the data in transformed and untransformed scale

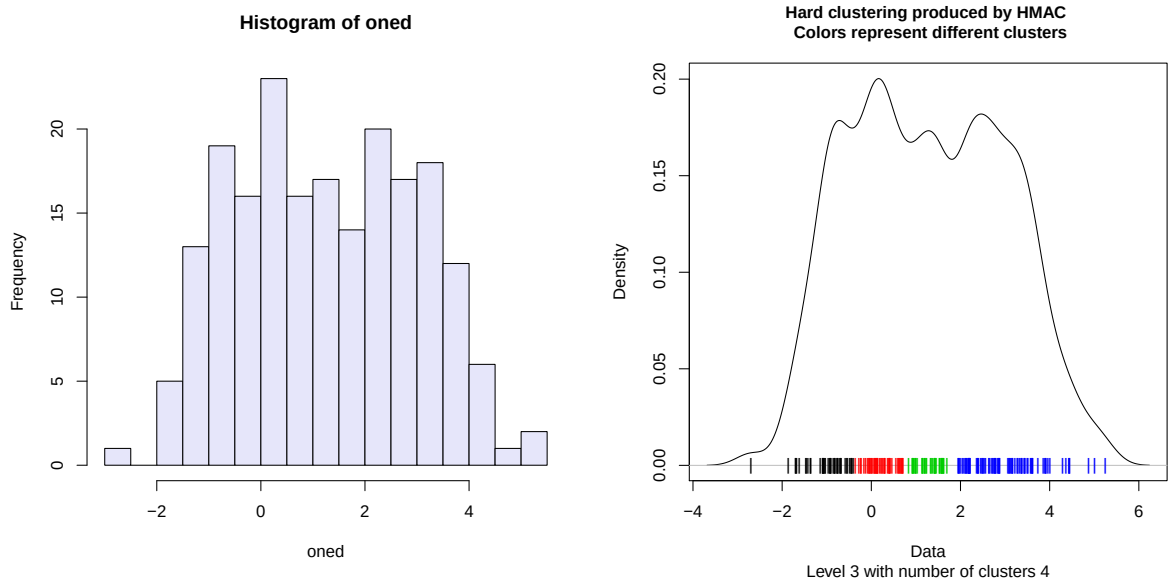
```
> data(logcta20.hmac)
```

```
> par(mfrow = c(1, 2))
> hard.hmac(cta20.hmac, n.cluster = 3)

There are no levels with exactly 3 clusters
The level at which there are 4 clusters is 4

> hard.hmac(logcta20.hmac, n.cluster = 3)

The level at which there are 3 clusters is 4

> par(mfrow = c(1, 1))
```



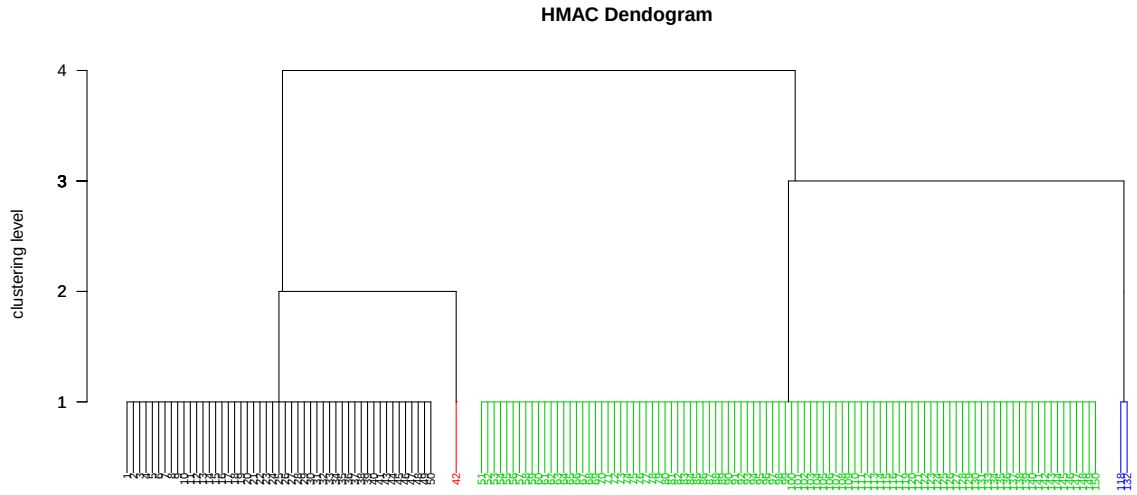
Now we demonstrate the output of the algorithm for a one dimensional example.

```
> data(oned.hmac)
> data(oned)
> par(mfrow = c(1, 2))
> hist(oned, n = 20, col = "lavender")
> hard.hmac(oned.hmac, level = 3)
> par(mfrow = c(1, 1))
```

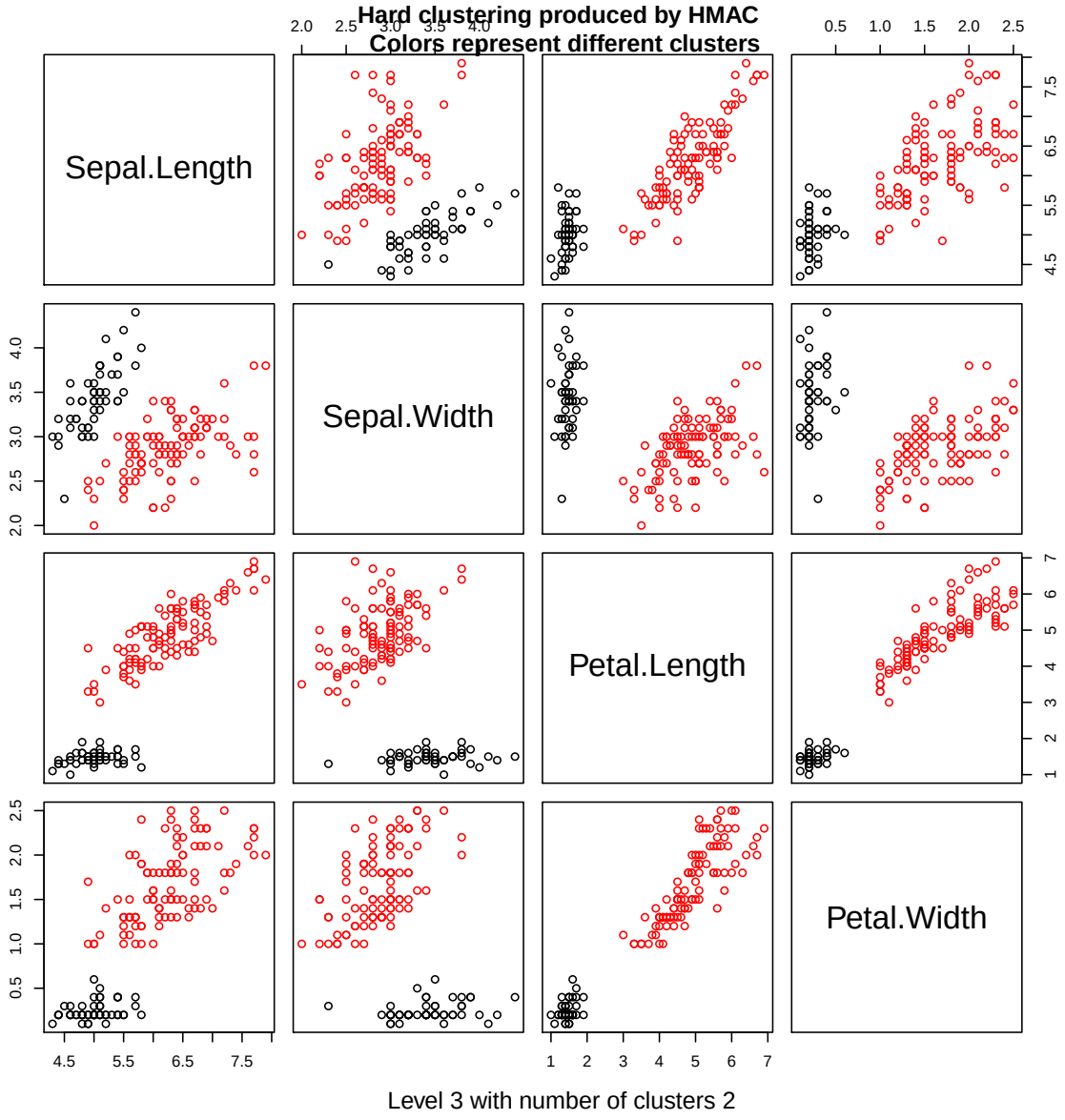


Finally we demonstrate the output of our algorithm for data with more than two dimensions The *iris* data from R datafile has five dimensions. The last column is categorical variable so we use the first four columns to extract the clusters in the data. The `plot` function will provide trees similar to two dimensional examples, but now the hard.hmac provides the pairs plot of the pairwise dimensions.

```
> iris.hmac = phmac(iris[, -5], npart = 1, parallel = F)

Performing initial Modal clustering..........
Building hierarchical Modal clusters at
level  1 ...level  2 ...level  3 ...level  4 ...level  5 ...
level  6 ...level  7 ...level  8 ...level  9 ...level  10 ...

> plot(iris.hmac, sep = 0.02)
```



```
> hard.hmac(iris.hmac, n.cluster = 2)

The level at which there are 2 clusters is 3
```



Acknowledgement

We thank Yeojin Chung and Jia Li for providing some of the initial codes for the *hmac* function used in this package

References

(i) Li. J, Ray. S, Lindsay. B. G, "A nonparametric statistical approach to clustering via mode identification," *Journal of Machine Learning Research*, 8(8):1687-1723, 2007.

(ii) Lindsay, B.G., Markatou M., Ray, S., Yang, K., Chen, S.C. "Quadratic distances on probabilities: the foundations," *The Annals of Statistics* Vol. 36, No. 2, page 983–1006, 2008.