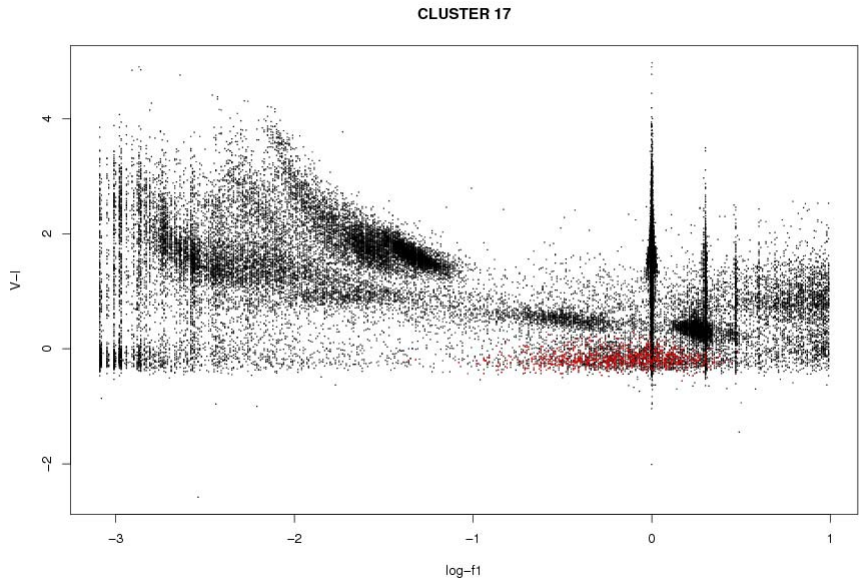
MASTER'S THESIS

Evaluation of unsupervised clustering algorithms for variable stars data



Pilar Herrera Plaza
Master thesis supervisor: Luis M. Sarro
UNED, 2008

Abstract

The aim of this master thesis is to assess the validity of unsupervised clustering algorithms to variable stars data classification for the Gaia mission. The use of these techniques allows to identify natural clustering without using any previous information about the classes and its distribution and, therefore, allows to discover new classes of objects. With this objective, we evaluate two probabilistic algorithms, one in which each cluster is characterized by a parametric distribution, and other, by a no-parametric distribution in a hierarchical clustering: Autoclass and HMAC (Hierarchical Mode Association Clustering). Both methods are evaluated against the same criteria, reproducibility, computation time, sensitivity to new classes and interpretability, in datasets that can grow up to 10^8 instances. These criteria are the first step to assess the feasibility of application of the algorithm but they are not enough to evaluate the goodness of clustering results. Despite the popular use of the unsupervised clustering techniques, the performance evaluation of clustering is an open question. It includes knowing how many clusters are actually present and how real is the clustering itself. Our clustering evaluation starts applying the expert knowledge and using a labeled dataset what allows to match some clusters with some variable stars types, but this is not enough to reach the objective of identifying each cluster. A review of the existing indices to evaluate clustering with objective criteria is included. Clusters and data are then analyzed to understand the results obtained with both methods biased by the method itself. A clustering combination method of these two algorithms is also tested as a technique that optimizes according multiple objective functions and trying to avoid some limitations of both algorithms.

Keywords: Unsupervised clustering, Autoclass, HMAC, model-based clustering, hierarchical clustering, validation indices.

Table of contents

1. Introduction

Clustering is a very popular technique of data mining and consists on the classification of data items into homogeneous groups that seem to fall naturally together. When classification operates under supervision by being provided information about training examples with the actual class, is called supervised classification. However, clustering is considered an unsupervised task because assumes no previous information about the data classes and its distribution and aims to discover it. The challenge is to find these clusters and assign instances to them. There are semisupervised variants of many clustering algorithms where prior knowledge can be incorporated into clustering algorithms but also can introduce any bias in the class discovery process.

There are several unsupervised clustering approaches that can be applied:

- **Partitioning algorithms** that construct various partitions and then evaluate them by some criterion. They are divided into major subcategories, the centroid and the medoids algorithms. The centroid algorithms represent each cluster by using the gravity center of the instances (k-means). The medoid algorithms represent each cluster by means of the instances closest to the gravity center (k-medoids).
- **Hierarchy algorithms** that create a hierarchical decomposition of the set of data (or objects) using some criterion. There are two major methods under this category: one is the agglomerative, the other the divisive. They use different criteria to decide which clusters shoud be merged or splitted. Under this category are: BIRCH, CURE, ROCK,..
- **Density-based** that are based on connectivity and density functions. They try to find clusters based on density of data points in a region. Some of these methods are DBSCAN, DENCLUE.
- **Grid-based** that first quantize the clustering space into a finite number of cells (hyperrectangles) and then perform the required operations on the quantized space. Cells that contain more than certain number of points are treated as dense and the dense cells are conected to form clusters. Some of them are: STING, WaveCluster and CLIQUE.
- **Model-based**: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other. Among the algorithms that are under this category are: Autoclass, SOM.

But not all clustering methods can adequately handle all sorts of cluster structures. And the evaluation of the goodness of clustering with objective criteria to select the one that better works for the data is a difficult task. So, the success of clustering is often measured subjectively in terms of how useful the results appear to be for the expert or by its ability to classify new cases.

1.1 Motivation

In this report, some unsupervised clustering algorithms are tested in large astrophysical datasets that gather information about variable starts for the GAIA mission.

Gaia is an ambitious mission planned by the European Space Agency ESA for 2011-2020 to chart a three-dimensional map of our Galaxy, the Milky Way, in the process revealing the composition, formation and evolution of the Galaxy. Gaia will provide unprecedented positional and radial velocity measurements with the accuracies needed to produce a stereoscopic and kinematic census of about one billion stars in our Galaxy.

Due to the great amount of data that will be generated, there is a consortium for analysis and processing of data generated, Data Processing and Analysis Consortium (DPAC). The pan-European consortium is organized into eight Coordination Units (CU) each of which is dedicated to a particular aspect of the full data processing task: CU1: System Architecture, CU2: Data Simulations, CU3: Core Processing, CU4: Object Processing, CU5: Photometric Processing, CU6: Spectroscopic Processing, CU7: Variability Processing, CU8: Astrophysical Parameters.

Within CU7 there is a working package called Object Clustering Analysis (OCA) with the aim of developing tools from the point of view of clustering. Although all the objects are well known in classes and subclasses for the scientific community, due to the great amount of objects, it is possible that new classes of objects can be observed by the first time in a quantity to be statistical significant. From May 2008 to April 2009, this coordination unit 7 will analyze several clustering algorithms to decide which one to adopt.

1.2 Contribution

This paper is a first contribution to the CU7 work. With this aim, we focused in two unsupervised methods that rely on statistical models: **Autoclass and HMAC** (Hierarchical Mode Association Clustering) that were tested in large astrophysical datasets.

Both methods differ in how is characterized each cluster and in its representation. In Autoclass, each cluster is characterized by a parametric distribution (multivariate Gaussian distribution for continuous data) whereas HMAC is characterized by a nonparametric distribution (Gaussian kernels for continuous data). About the representation, Autoclass allows one instance to belong to more than one cluster in a probabilistic way (finite mixtures) and

HMAC produces a bottom-up hierarchical structure, starting with each instance being a cluster and finishing when all instances are grouped in a unique cluster in a diagram called dendrogram.

The study consists on assessing the feasibility of application of both methods according to some important criteria for this working group: reproducibility, computing time, sensitivity to new classes and interpretability. The clustering evaluation according to some objective criteria that allows to compare them is also undertaken but only to show its extreme difficulty. Finally, from a detailed analysis of resulting clusters is also possible to extract some knowledge about data, the probability density model that better fits, and, consequently, to suggest other methods with more chances to success.

As a result of the scientific validation of Autoclass, the report "Assessment of the validity of Autoclass for CU7 unsupervised classification" Ref: GAIA-C7-TN-SVO-LSB-012-D (draft version) was accepted by the GAIA consortium.

1.3 Organization

This report is organized as follows. Chapter 2 is a theoretical review of both proposed clustering algorithms, Chapter 3 explains the datasets used and the analysis method, Chapter 4 and 5 show the results of applying Autoclass and HMAC respectively. Chapter 6 explains a clustering combination method using both algorithms. In Chapter 7 it is shown the difficulty to evaluate clustering, and this is done mainly under subjective criteria, and to finish, Chapter 8 gathers the final conclusions.

2. Unsupervised clustering algorithms: Autoclass and HMAC

This chapter gives a brief theoretical background of both selected methods to process large astrophysical datasets and points out some differences between them.

2.1 Autoclass

Autoclass is an unsupervised clustering algorithm based on Bayesian classification. The goal is to find the most probable class descriptions given the data and prior expectations. This approach is an alternative method to maximum likelihood approach that tries to find the class descriptions that best predict data. Whereas maximum likelihood estimation favours models with many parameters and many number of classes (until the classes equals the number of data points), the Bayesian classification enforces a trade-off between the fit to the data and the complexity of the class descriptions. This is done considering that each simple parameter introduced into a Bayesian model brings its own multiplicative prior to the joint probability that always lowers the marginal. If a parameter fails to raise the marginal by increasing the direct probability by a greater factor than the prior lowers the marginal, the model incorporating that parameter is rejected. The introduction of these priors always favours classifications with smaller number of classes and avoids overfitting.

Autoclass requires to be explicit about the space of models, a parametrized probability distribution of density function, one is searching in. The simplest model to predict is when each attribute is independent. For real attributes, one can use a standard normal distribution. But Autoclass can also deal with attributes that are not independent of each other in each class and can be correlated following a multivariate normal distribution. For a set of real continuous attributes, this means that a new set of attributes can be defined as linear combinations of the ones given, which vary independently according to normal distributions. Simple independent attribute models require fewer parameters than the corresponding covariant models.

Autoclass follows the classical mixture model. In this method, class membership of each instance is expressed probabilistically. The cases are not assigned to a class but each case has a probability of being member of different classes, this is the interclass mixture probability, and the sum of these probabilities must be one.

When the probabilities of most instances in a cluster are around 0.99 in this most probable class, we can assume that we are dealing with a well separated cluster. On the contrary, clusters characterized by instances with probability vectors lower than 0.5 in all classes, are low contrast clusters that can be assumed to be overlapped with other distributions. One can has, in addition to the clustering, a measure of how well the classification fits the data and the individual data fit the classes.

How Autoclass works

Autoclass input is a set of data instances, a model class (normal or multivariate normal) over continuous real attributes and a set of search parameters to configure the algorithm: number of classes to start, random initializations, stopping criterion...

The process of knowledge discovery is an iterative process. Autoclass repeatedly creates a random classification and then computes the probabilistic class memberships of data instances using the class parameters and the implied relative likelihoods. Using the new class members computes class statistics (mean and variance or covariances) and revises class parameters. This process is repeated until it converges to some local maximum (the clusters stop changing or the change is less than a threshold for several consecutive iterations). Each iteration is called a ”try”.

Autoclass output is a set of the best classifications found. The most important indicator of the relative merits of these classification is the log total posterior probability value. As the probability is between 1 and 0 the log probability is negative from 0 to negative infinite, and it is expressed as follows: PROBABILITY exp(-1727410.430).

A classification is composed of:

- a set of classes described by a set of parameters (mean, covariance matrix for correlated model), which specify how the class is distributed along the various attributes. It gives also the influence of each attribute in the classification.
  e.g.
  Attr1 in class 0 (Mean,StDev)= (-1.46e+00 +1.64e-01)
  Attr1 in all datasets (Mean,StDev)= (-9.46e-01 +1.19e+00)
  Attr1 attribute influence in class 0 = +3.13e+00, computed as the difference of means of both distributions divided by the standard deviation of the attribute in the class.

- a set of class weights describing what percentage of cases are likely to be in each class. This is not the number of instances belonging to a class but the sum of membership probabilities for that class. Autoclass also gives other parameters as the class strength or cross entropy.
  e. g.
  Class 0, Class weight 5199, normalized class weight 0.120
  Class 0, Log of class strength = -3.70e+001 Relative Class strength 2.75e-005, computed as the geometric mean probability that any instance belongs to a class.
  Class 0, class cross entropy with respect to global class 7.25e+000, defined as divergence between two probability distributions. It ranges

form zero for identical distributions to infinite for distributions placing probability 1 on differing values of an attribute.

- a probabilistic assignment of cases to the classes:
  Case N (Class Prob) (Class Prob) (Class Prob)
  001 (12 0.964) (25 0.034) (11 0.002)

2.2 HMAC- Hierarchical Mode Association Clustering

HMAC is another unsupervised clustering algorithm that is based on finding local maxima (mode identification) of mixture densities by applying a nonparametric density estimate. A cluster is formed by those points that ascend to the same local maximum of the density function. The major advantage of this approach is that, without a model fitting, it yields a density description for every cluster.

The path from a point to its associated mode, a local maximum of density, is solved by an algorithm called MEM (Modal Expectation Maximization) similar to EM algorithms but with the objective of finding the local maxima, modes, of a given distribution.

In the implementation of HMAC used in this report, the non parametric kernel density estimation uses a Gaussian kernel for continuous data, having a spherical covariance matrix with a standard deviation σ (bandwidth) in the diagonal, which is equivalent to modeling the different variables separately.

The use of this non parametric kernel does not determine the clustering. This approach can be used to find the modes of any density in the form of a mixture distribution because mixture components only play the role of approximating a density. The approach is robust even when clusters deviate substantially from Gaussian distributions.

The algorithm also supplies a pairwise separability measure for clusters defined using the ridgeline between the modes of two clusters. Using this separability measure, it supplies a merging method for clusters weakly separated.

This approach is an agglomerative hierarchical process. First we show how is the modal clustering in a level, and then we show the extension to hierarchical version.

Modal Clustering

Modal clustering comprises the following steps:

Let the set of data to be clustered be $S = \{x_1, x_2,....,x_n\}$ $x_i \in R^d$ .The Gaussian kernel density estimate is formed:

$$f(x) = \sum_{i=1}^{n} \frac{1}{n} \phi(x \mid x_i, \Sigma)$$

where the Gaussian density function

$$\phi(x \mid x_i, \Sigma) = \frac{1}{(2\pi)^{d/2} \mid \Sigma \mid^{1/2}} \exp(-\frac{1}{2}(x - x_i)^t \Sigma^{-1}(x - x_i))$$

Σ is a spherical covariance matrix $\Sigma = \text{diag}(\sigma^2; \sigma^2; ..., \sigma^2) = D(\sigma^2)$. The standard deviation σ is also referred to as the bandwidth of the Gaussian kernel.

With a given Gaussian kernel covariance matrix $D(\sigma^2)$, data are clustered as follows:

1. Form kernel density
2. Use $f(x \mid S, \sigma^2)$ as the density function. Use each x_i, $i = 1,2, ...,n$, as the initial value in the MEM algorithm to find a mode of $f(x \mid S, \sigma^2)$. Let the mode identified by starting from x_i be $M\sigma(x_i)$.
3. Extract distinctive values from the set $\{M\sigma(x_i); i = 1;2; ..;n\}$ to form a set G. Label the elements in G from 1 to $|G|$. In practice, due to finite precision, two modes are regarded equal if their distance is below a threshold.
4. If $M_\sigma(x_i)$ equals the kth element in G, x_i is put in the kth cluster.

Hierarchical modal clustering

Hierarchical approach starts with every point x_i being a cluster by itself. When the variances of kernel increase, the density estimate becomes smoother and tends to group more points in one cluster. Hierarchical clustering is performed in a bottom-up manner. In each hierarchical level, modes acquired at a smaller bandwidth are treated as points to be clustered when a larger bandwidth is used. A hierarchy of clusters can be thus constructed by gradually increasing the variances of Gaussian kernels.

Mechanisms for merging clusters

Merging clusters can be produced when increasing the bandwidth value. However, as this can cause that prominent clusters be clumped while leaving small clusters unchanged, this approach supplies two additional mechanisms for merging.

One method uses a separability measure between two clusters based on the ridgelines, lines that join bumps of two clusters, which takes comprehensive consideration of the exact densities of the clusters. Merging clusters according to this parameter can avoid the aforementioned problem. The idea is to absorb other clusters when they are not well separated or are less dominant.

Other method has into account outliers. Outliers are points that are far from all essential clusters and tend to have high separability so they will not be merged with the previous mechanism. HMAC uses the parameter called "coverage rate" to define if merge outliers. The coverage rate define if a cluster is an outlier based on its size in proportion to the total dataset. Outliers will be merged to other clusters.

2.3 Differences between both methods

There are clear differences between Autoclass and HMAC. In Autoclass, each cluster is characterized by a parametric distribution (Normal or multivariate Gaussian distribution for continuous data) whereas in HMAC each cluster is found using a nonparametric distribution (Gaussian kernels for continuous data). In HMAC, there is not a model fitting.

The representation of results is also different. Autoclass allows one instance to belong to more than one cluster in a probabilistic way and HMAC produces a bottom-up hierarchical structure, starting with each instance being a cluster and finishing when all instances are grouped in an unique cluster in a diagram called dendrogram.

This representation produces a new difference. Autoclass supplies the final number of classes, but HMAC does not. Each hierarchical level has its own number of classes and it is necessary to incorporate expert knowledge to decide which hierarchical level to select.

However, HMAC has some interesting advantages against Autoclass. These are the irrelevance of initialization and the easiness of implementing required optimization techniques. In addition, with HMAC approach each cluster accounts for a distinct hill or mode of the probability density, fact that not happens with parametric approaches when dealing with mixture distributions.

3. Data description and analysis method

In this section, we review the astrophysical datasets used for the study and the attributes that they contain, show some plots of location of instances in several attributes, and define the criteria to evaluate the algorithms.

3.1 Astrophysical datasets description

To perform our experiments we used the **OGLE Large Magellanic Cloud dataset** with 43351 cases of variable stars without any information about the classes they belong to. The instances consisted of thirteen real attributes (with names and explanations listed in Table 3.1) with no missing values. We also used the **Hipparcos dataset** (2498 instances) with exactly the same attribute information as the original OGLE dataset.

For Autoclass experiments, we assumed that attributes followed a normal distribution and that all of them could be correlated, which implied a multivariate normal distribution.

Attribute	Meaning
log-f1	log of the first frequency
log-f2	log of the second frequency
log-af1h1-t	log amplitude first harmonic first frequency
log-af1h2-t	log amplitude second harmonic first frequency
log-af1h3-t	log amplitude third harmonic first frequency
log-af1h4-t	log amplitude fourth harmonic first frequency
log-af2h1-t	log amplitude first harmonic second frequency
log-af2h2-t	log amplitude second harmonic second frequency
log-crf10	amplitude ratio between harmonics of the first frequency.
pdf12	phase difference between harmonics of first frequency
varrat	variance ratio before and after 1st frequency subtraction
B-V	colour index
V-I	colour index

Table 3.1. Dataset attributes.

We also used some labeled datasets to validate the obtained clustering results and to help to identify the clusters. These datasets were taken from published results on the OGLE database and belong to the following classical variable stars: Cepheids (1313 cases - cep), double mode Cepheids (71 - dmcep), eclipsing binaries (2467 - ecl), eclipsing binaries (Groenewegen sample; 162 - new-ecl), ellipsoidals-eclipsing binaries (80 - ell-ecl), ellipsoidals (613 - ell-ell), long period variables (Mira and Semirregular variables; 2735 - lpv), PT Cep (14 ptcep), RR Lyrae stars (types AB and C; 2558 - rrlyr) and double mode RR Lyrae stars (50 - rrd).

Another dataset of multiperiodic variables stars data were taken from the classification carried out by the hierarchical classifier presented in Sarro et al. (2008), and include β Cephei stars (292 cases - all.2.bcep.8), δ Scuti stars (22 - all.2.dscut.8), γ Doradus (102 - all.2.gdor.8), Pulsating Variable Super Giants (PVSG; 79 - all.2.pvsg.7), and Slowly Pulsating B stars (590 - all.2.spb-8).

Data were supplied previously processed and 9 from 13 attributes presented a logarithmic transformation, which effect was also investigated. This transformation has its importance since Autoclass is a model fitting based algorithm and the model selected is a multivariate normal distribution. If data under the log-transformation do not behave normally or deviate considerably of a normal distribution, the fit can conduce to unexpected results. The same happens with HMAC algorithm that uses a Gaussian kernel to model data.

However, due the nature of the attributes under this transformation (frequencies, amplitude of harmonics) with values greater than 0 it can be expected a positively skewed distribution that could be modeled by the log-normal distribution. If in this case, this preprocessing step could improve the clustering. If not, there is also the possibility of being in the case of normal distributions with small sigma values (< 0.1) where the log-normal is visually indistinguishable from a normal.

3.2 Algorithms implementation.

Both algorithms were already implemented and were taken directly from their authors. We selected versions in C instead on R because both algorithms are extremely computation intensive and C executes up to 20 times faster than R.

Autoclass version used was autoclass-c-win-3-3-4 compiled with Microsoft Visual C++ 6.0, its original version.

HMAC is presented in three versions: *mtree*, *mtreeridge* and *mtreesep* that are also coded in C.

- **mtree** is the basic version that performs the hierarchical modal association clustering (HMAC), and outputs the clustering results at each level of the hierarchy as well as the dendrogram created.

- **mtreeridge** performs the same clustering procedure first, then computes the ridgeline and the density value along the ridgeline line for either a given pair of clusters at a given level, or for all the pairs of clusters at all the levels.

- **mtreesep** performs the same clustering procedure first, then conducts the separability and coverage rate based merging at either a given level, or all the levels of the dendrogram.

HMAC versions were compiled with Microsoft Visual C++ 6.0. But previous to our tests we had to correct a code bug (related to memory deallocation) that did not allow to process more than 10500 instances with13 attributes. Further in this report, other memory problems with this code will arise.

Both algorithms were run on a computer with a Corel Duo T7500 processor and 2 GB RAM memory although the native implementation of these algorithms are not able to use the double core and the percentage of use of the microprocessor when they are running is only 50%. The O.S. was Windows Vista Premium.

3.3 Plots of datasets

The best way to analyze data is to plot the location of instances in 2-D projections of pairs of attributes. In this section we only show a selection of plots of OGLE (Figure 3.1) and Hipparcos datasets (Figure 3.2) in several attributes. In OGLE dataset, there is an evident presence of data artifacts in attributes log-f1 and log-f2 that, undoubtedly, will affect clustering.

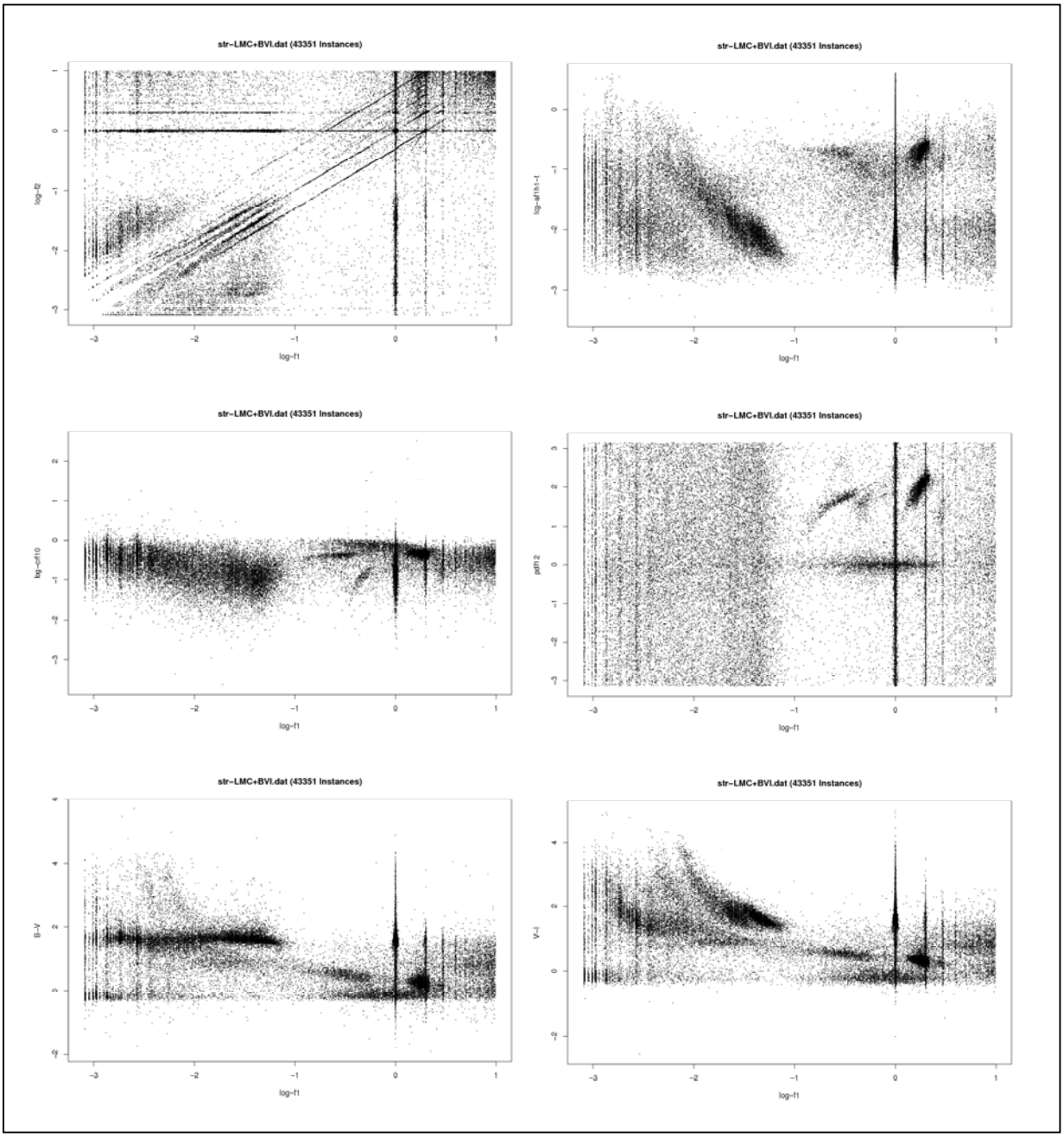


Figure 3.1 - Plot of attributes log-f1 vs log-f2, log-af1h1-t, log-crf10, pdf12.B-V and V-I of OGLE dataset.

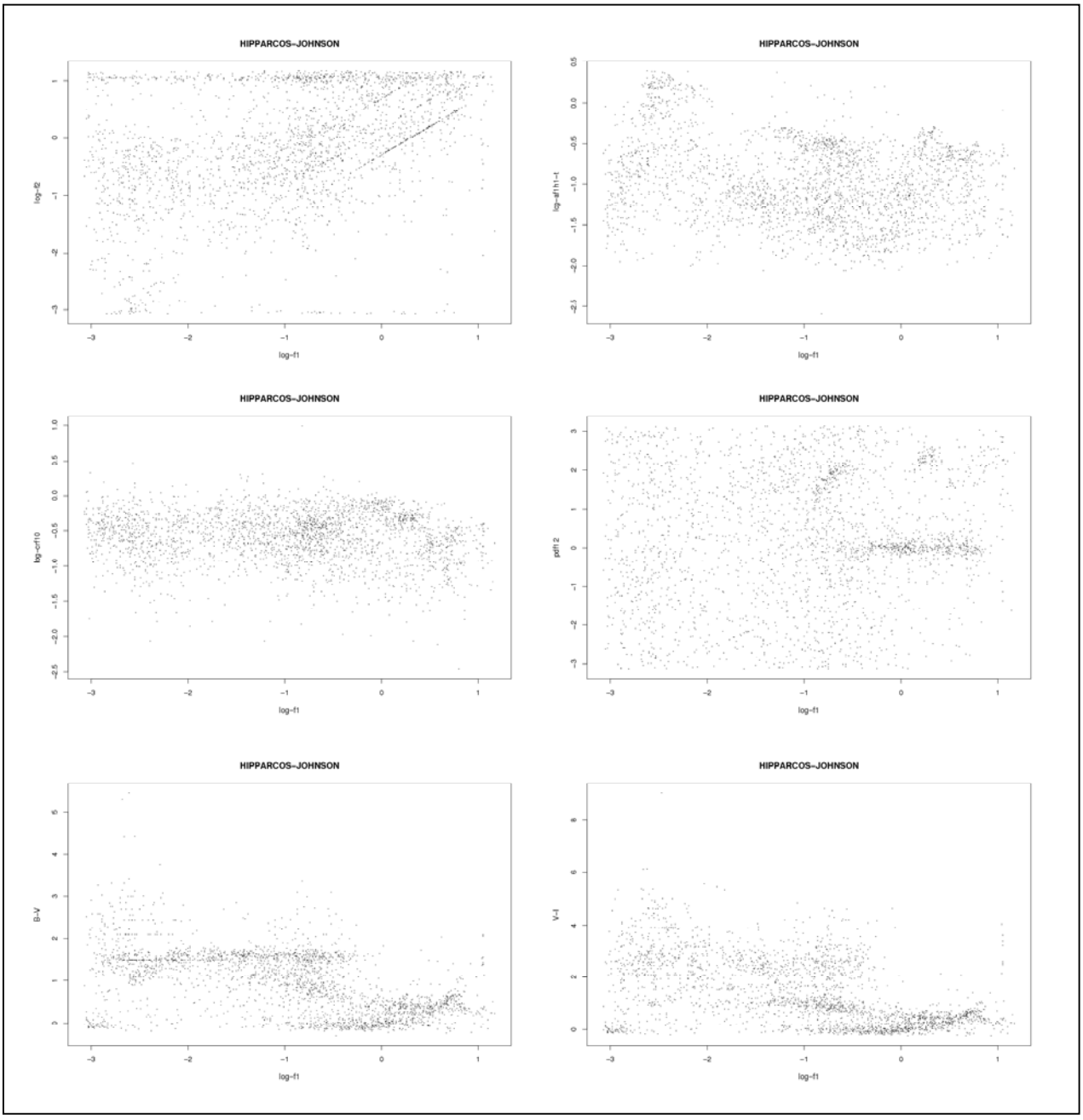


Figure 3.2 - Plot of attributes log-f1 vs log-f2, log-af1h1-t, log-crf10, pdf12, B-V and V-I of Hipparcos dataset.

The visual analysis over both datasets foresee that clustering over OGLE dataset can be easier than over Hipparcos dataset because the higher density of instances of it.

In the process of cluster identification, we had available some labeled datasets. In order to try to understand the results, the labeled datasets are also plot in the

same attributes (Figure 3.3). Each variable star type is plot in a different color. Spurious values in attributes log-f1 and log-f2 are also present in these data.

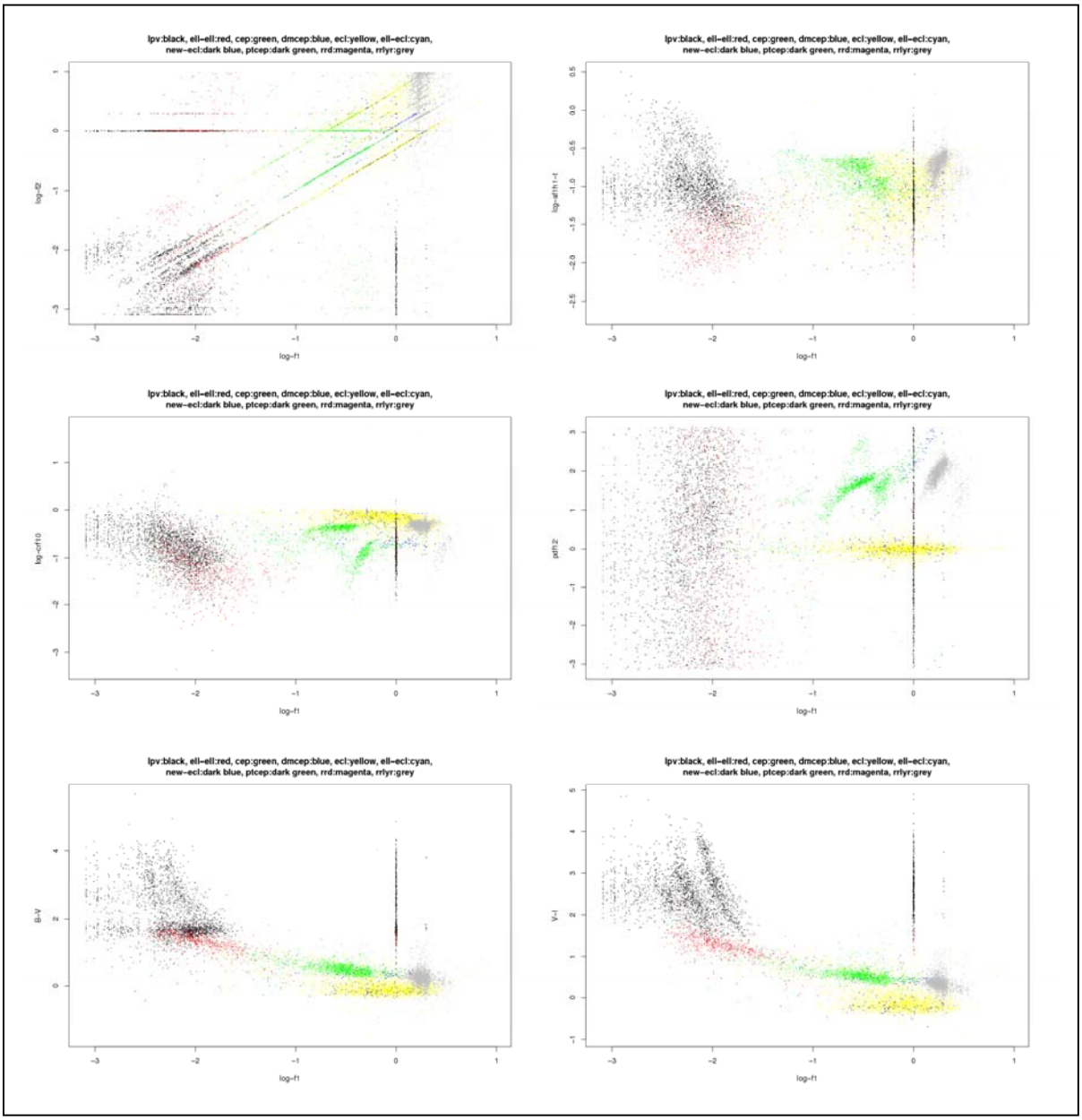


Figure 3.3 - Plots of attributes log-f1 vs log-f2, log-af1h1-t, log-crf10, pdf12.B-V and V-I of labeled datasets.

Finally, we show some plots (Figure 3.4) with the attributes without the log-transformation that already makes think about the suitable of this transformation due to the highly skewed distribution.

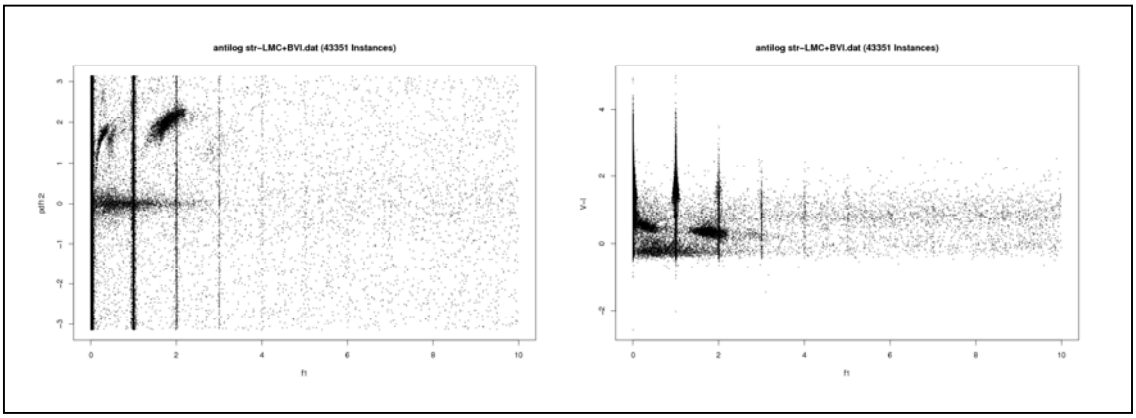


Figure 3.4 - Plot of attributes f1 vs pdf12 and V-I of OGLE dataset.

3.4 Criteria to evaluate clustering algorithms

To investigate the feasibility of applying Autoclass and HMAC to the clustering of variable stars data in the context of CU7, Autoclass and HMAC were assessed attending to the following criteria: reproducibility, computation time, sensitivity to new classes and interpretability.

The selection of these criteria comes from the fact that the clustering algorithms will be applied to some extremely large datasets and there must be a reasonable limit to the total time to process them. About the reproducibility, it is necessary that consecutive clustering during the mission give similar results. Algorithms must also be able to discover new classes, criteria that was tested by introducing some synthetic instances in the dataset and verifying if the algorithm was able to find them. Finally, the resulting clusters should have an intuitive relationship with known variability classes in order to the clustering has any utility.

4. Results of application of AUTOCLASS

In this chapter, the results of application of Autoclass are presented.

The first decision to apply Autoclass was to select the model that the data must fit. As we assumed that attributes followed a normal distribution (after $\log_{10}$-transformation of the original data in most attributes) and that all of them could be correlated, the hypothesized model was a multivariate normal distribution. This covariant model leads to cases being classified by changes in the relation between the covarying attributes rather than by absolute differences in their values.

AutoClass also requires to specify measurement uncertainty for continuous variables. Data that are more precise have more influence on the final classification. Here, all attributes were considered equally precise (0.05).

4.1 Impact of randomness in clustering results.

The first criterion to evaluate Autoclass is the impact of random initializations in clustering results when reproducible results are needed.

Autoclass can be configured to have random initializations or not. To verify the effect of these random initializations we specify the following parameters in Autoclass search: *force_new_search_p = true, start_fn_type = "random", randomize_random_p = true*.

We asked the system to find the best clustering starting with different number of clusters (*start_j_list = 20,30,40,50,60*), 24 hours of calculi (*max_duration = 86400*) and all attributes following a multivariate normal distribution (*multi_normal_cn 0 1 2 3 4 5 6 7 8 9 10 11 12*).

This process was repeated 5 times with random initializations over the total dataset (43351 cases). Table 4.1 shows the five best results obtained from each run. As can be observed, the number of clusters differs significantly for each one (75, 66, 54, 58, 93). However, this result is not enough to decide if the results are very different. This is so because discrepancies could possibly reduce to the numerous small diffuse clusters that gather noisy detections. In order to evaluate the discrepancies among clustering results, we analyzed the instances composition of the main clusters and tried to match clusters in the runs with the highest and lowest numbers of clusters (5 and 3 respectively).

The process of matching different clusters gave the following results:

- the clusters of greater weight in run 3 can easily be found on run 5 with a 95-100 % overlap of instances.

- some clusters on run 3 could be identified as split into two or three clusters on run 5.

- there were clusters on run 3 spread into many run 5 clusters with a maximum percentage less than 50%.

This last case implies that the results of Autoclass depend on random initializations negatively.

Nevertheless, although the clustering process gives different results depending on the initialization, astrophysical meaningful clusters (see Section 4.5) can be identified in both runs. In some cases, the mixture of instances in clusters (third case) takes place between clusters with the same astrophysical interpretation thus revealing weak support for the separation in clusters and proving that the variability induced by the random initialization is less harmful than expected.

Run	Results					
1	PROBABILITY	exp(-1720621.160)	N_CLASSES	75	FOUND ON TRY	55
	PROBABILITY	exp(-1721327.600)	N_CLASSES	79	FOUND ON TRY	24
	PROBABILITY	exp(-1721964.400)	N_CLASSES	76	FOUND ON TRY	87
	PROBABILITY	exp(-1722131.330)	N_CLASSES	81	FOUND ON TRY	68
	PROBABILITY	exp(-1722543.580)	N_CLASSES	78	FOUND ON TRY	98
2	PROBABILITY	exp(-1722692.460)	N_CLASSES	66	FOUND ON TRY	125
	PROBABILITY	exp(-1723144.750)	N_CLASSES	58	FOUND ON TRY	63
	PROBABILITY	exp(-1723567.070)	N_CLASSES	63	FOUND ON TRY	64
	PROBABILITY	exp(-1723848.910)	N_CLASSES	60	FOUND ON TRY	66
	PROBABILITY	exp(-1724399.960)	N_CLASSES	59	FOUND ON TRY	142
3	PROBABILITY	exp(-1721932.640)	N_CLASSES	54	FOUND ON TRY	116
	PROBABILITY	exp(-1722686.420)	N_CLASSES	50	FOUND ON TRY	80
	PROBABILITY	exp(-1723819.880)	N_CLASSES	49	FOUND ON TRY	49
	PROBABILITY	exp(-1724247.280)	N_CLASSES	57	FOUND ON TRY	155
	PROBABILITY	exp(-1724682.560)	N_CLASSES	45	FOUND ON TRY	37
4	PROBABILITY	exp(-1721934.470)	N_CLASSES	58	FOUND ON TRY	66
	PROBABILITY	exp(-1722213.260)	N_CLASSES	63	FOUND ON TRY	80
	PROBABILITY	exp(-1722794.120)	N_CLASSES	67	FOUND ON TRY	128
	PROBABILITY	exp(-1723124.660)	N_CLASSES	59	FOUND ON TRY	95
	PROBABILITY	exp(-1723219.280)	N_CLASSES	63	FOUND ON TRY	107
5	PROBABILITY	exp(-1721488.460)	N_CLASSES	93	FOUND ON TRY	69
	PROBABILITY	exp(-1722657.520)	N_CLASSES	88	FOUND ON TRY	68
	PROBABILITY	exp(-1722714.380)	N_CLASSES	96	FOUND ON TRY	95
	PROBABILITY	exp(-1722881.730)	N_CLASSES	86	FOUND ON TRY	98
	PROBABILITY	exp(-1722887.470)	N_CLASSES	86	FOUND ON TRY	86

Table 4.1: Five best results (ranked according to the marginal joint probability) found in each of the 5 runs with random initializations. The table includes the number of cluster in each solution and the try number when this solution was found.

4.2 Impact of computation time on clustering results

The second criterion to evaluate Autoclass is the impact of computation time on clustering results. It is important to ensure that the best clustering is somehow

robust to an increase in the computation time. The objective of the experiments is to quantify how different are two 'best' solutions found differing only in the time allowed to find them.

This time we run Autoclass to find the best solution after 6, 12, 24 and 48 hours of processing. We configured Autoclass without random initialization for repeatable results (*force_new_search_p= true*, *start_fn_type = "block", randomize_random_p = false*), starting with different number of clusters (*start_j_list = 20,30,40,50,60*), and all attributes following a multivariate normal distribution (*multi_normal_cn 0 1 2 3 4 5 6 7 8 9 10 11 12*). Autoclass was applied over the total dataset (43351 cases).

Table 4.2 shows the results obtained. In all cases, the best solution was found on try 35, on the first 6 hours.

Run	Results
6h	PROBABILITY exp(-1727410.430) N_CLASSES 48 FOUND ON TRY 35
	PROBABILITY exp(-1727727.270) N_CLASSES 48 FOUND ON TRY 33
	PROBABILITY exp(-1729959.700) N_CLASSES 51 FOUND ON TRY 29
	PROBABILITY exp(-1730302.420) N_CLASSES 48 FOUND ON TRY 37
	PROBABILITY exp(-1730347.620) N_CLASSES 55 FOUND ON TRY 25
	SEARCH SUMMARY 43 tries over 6 hours 11 minutes 57 seconds
12h	PROBABILITY exp(-1727410.430) N_CLASSES 48 FOUND ON TRY 35
	PROBABILITY exp(-1727727.270) N_CLASSES 48 FOUND ON TRY 33
	PROBABILITY exp(-1729902.890) N_CLASSES 51 FOUND ON TRY 48
	PROBABILITY exp(-1729925.720) N_CLASSES 51 FOUND ON TRY 71
	PROBABILITY exp(-1729959.700) N_CLASSES 51 FOUND ON TRY 29
	SEARCH SUMMARY 82 tries over 12 hours 5 minutes 10 seconds
24h	PROBABILITY exp(-1727410.430) N_CLASSES 48 FOUND ON TRY 35
	PROBABILITY exp(-1727727.270) N_CLASSES 48 FOUND ON TRY 33
	PROBABILITY exp(-1727775.850) N_CLASSES 48 FOUND ON TRY 135
	PROBABILITY exp(-1728142.430) N_CLASSES 48 FOUND ON TRY 156
	PROBABILITY exp(-1728449.990) N_CLASSES 48 FOUND ON TRY 118
	SEARCH SUMMARY 163 tries over 1 day 39 seconds
48h	PROBABILITY exp(-1727410.430) N_CLASSES 48 FOUND ON TRY 35
	PROBABILITY exp(-1727431.700) N_CLASSES 48 FOUND ON TRY 321
	PROBABILITY exp(-1727450.590) N_CLASSES 48 FOUND ON TRY 314
	PROBABILITY exp(-1727467.710) N_CLASSES 48 FOUND ON TRY 223
	PROBABILITY exp(-1727571.550) N_CLASSES 48 FOUND ON TRY 228
	SEARCH SUMMARY 308 tries over 2 days 12 minutes 32 seconds

Table 4.2: Five best results (ranked according to the marginal joint probability) found in each of the 5 runs with increasing allowed computation time (6, 12, 24, 48 hours). The table includes the number of cluster in each solution and the try number when this solution was found.

In this particular case, the interpretation of the best clustering does not change simply because the best clustering is found in the first block of the experiment. So the test was repeated four more times with random initializations and 48 hours of processing. We also changed the initial values of clusters to test (start j list) to induce more variability. As time increases linearly with the tries, from the best try we could estimate the time needed to find it (Table 4.3).

Run	Results
48h	PROBABILITY exp(-1722177.170) N_CLASSES 55 FOUND ON TRY 34 PROBABILITY exp(-1722182.650) N_CLASSES 62 FOUND ON TRY 179 PROBABILITY exp(-1722711.210) N_CLASSES 61 FOUND ON TRY 143 PROBABILITY exp(-1723109.090) N_CLASSES 62 FOUND ON TRY 245 PROBABILITY exp(-1723347.480) N_CLASSES 66 FOUND ON TRY 70 SEARCH SUMMARY 261 tries over 2 days 4 minutes 16 seconds start_j_list = 20,30,40,50,60
48h	PROBABILITY exp(-1723167.200) N_CLASSES 57 FOUND ON TRY 221 PROBABILITY exp(-1723771.670) N_CLASSES 65 FOUND ON TRY 293 PROBABILITY exp(-1723948.090) N_CLASSES 57 FOUND ON TRY 93 PROBABILITY exp(-1723991.140) N_CLASSES 55 FOUND ON TRY 208 PROBABILITY exp(-1723991.410) N_CLASSES 58 FOUND ON TRY 207 SEARCH SUMMARY 308 tries over 2 days 13 minutes 9 seconds start_j_list = 20,25,30,35,50
48h	PROBABILITY exp(-1731024.540) N_CLASSES 38 FOUND ON TRY 97 PROBABILITY exp(-1731122.000) N_CLASSES 44 FOUND ON TRY 394 PROBABILITY exp(-1731278.640) N_CLASSES 41 FOUND ON TRY 128 PROBABILITY exp(-1731392.520) N_CLASSES 41 FOUND ON TRY 70 PROBABILITY exp(-1731392.870) N_CLASSES 41 FOUND ON TRY 387 SEARCH SUMMARY 421 tries over 2 days 2 minutes 24 seconds start_j_list = 2,4,8,16,32
48h	PROBABILITY exp(-1721983.160) N_CLASSES 69 FOUND ON TRY 44 PROBABILITY exp(-1722087.630) N_CLASSES 71 FOUND ON TRY 185 PROBABILITY exp(-1722092.290) N_CLASSES 65 FOUND ON TRY 68 PROBABILITY exp(-1722151.840) N_CLASSES 66 FOUND ON TRY 46 PROBABILITY exp(-1722569.470) N_CLASSES 66 FOUND ON TRY 110 SEARCH SUMMARY 213 tries over 2 days 6 minutes 12 seconds start_j_list = 21,32,43,54,65

Table 4.3: Five best results (ranked according to the marginal joint probability) found in each of the 5 runs with random initializations and 48 hours of computation time. The table includes the number of cluster in each solution and the try number when this solution was found.

Table 4.4 summarizes these results.

Run	Clusters	Try	Total tries	Time estimation	Probability
1	48	35	308	6h	exp(-1727410.430)
2	55	34	261	12h	exp(-1722177.170)
3	57	221	308	36h	exp(-1723167.200)
4	38	97	421	12h	exp(-1731024.540)
5	69	44	213	12h	exp(-1721983.160)

Table 4.4. Comparison of different runs and time estimation of the best result according to the marginal joint probability.

The results show that the best solution can be obtained very early in the processing and that a reasonable increase in computation time does not imply a better solution according to the log posterior probability value of the classification under the Autoclass assumptions. Again, the results are conditioned by the initialization parameters.

4.3 Impact of dataset size on computation time

The next investigated aspect is the increment of processing time with increasing dataset size, assuming that the algorithm must be able to handle the order of 10^8 instances.

In order to perform our experiments we took the original dataset of 43351 cases and we duplicated instances to get datasets of 50000, 75000, 100000, 200000, 500000, 1000000 cases. Autoclass was configured to find a solution over each dataset with the following parameters: no random initializations (*force_new_search_p = true, start_fn_type = "block", randomize_random_p = false*), a fixed number of iterations (*max_n_tries = 50*), starting with different number of clusters (*start_j_list = 20,30,40,50,60*), and all attributes following a multivariate normal distribution (*multi_normal_cn 0 1 2 3 4 5 6 7 8 9 10 11 12*). The results were meaningless (see Table 4.5) when the dataset increased its size over 100000 instances.

Instances	Processing time (secs)	Clusters found
50000	28337	58
75000	47183	56
100000	77696	49
200000	5573	2
500000	17342	4
1000000	28149	8

Table 4.5: Time needed to perform 50 iterations with different datasets sizes and number of clusters characterizing the most probable solution. The table illustrates the problem with increasing number of instances.

We created other datasets with sizes between 100000 and 200000 instances to find the exact point where the processing time reduced. We found (Table 4.6) that the critical point is between 129000 and 130000 instances but solutions started to make no sense from 125000 instances. Although the run with 129000 instances had 50 clusters, closer examination of these clusters proved the existence of a single cluster with a weight of 99.8 %.

Instances	Processing time (secs)	Clusters found
100000	77696	49
125000	114767	1
128000	100235	130
129000	243110	50
130000	4869	1
131000	7193	1
150000	3854	1
200000	5573	2

Table 4.6: Time needed to perform 50 iterations with datasets sizes between 100000 and 200000 instances.

We tried another two different configurations with 129000 instances: random initializations (*force_new_search_p = true, start_fn_type = "random", randomize_random_p = true*), other starting list of clusters (*start_j_list = 60,100,140,200,300*), maintaining a fixed number of iterations (*max_n_tries = 50*). Both configurations gave, as the best solution, 1 cluster. Most interesting, we also tried a different simpler model (*single_normal_cn*) with many less parameters and the results were again in agreement with the density structure of the data points.

Finally, we tried a classification using only 5 attributes and the multivariate normal distribution. We found that, for 129000 instances, Autoclass could find a sensible solution. However the problem reappeared again when Autoclass had to process 500000 instances: premature convergence and small number of clusters (Table 4.7).

```
AutoClass CLASSIFICATION for the 129000 cases
SEARCH SUMMARY 50 tries over  9 hours 24 minutes 52 seconds

PROBABILITY  exp(-2419278.140) N_CLASSES  79 FOUND ON TRY   23 *SAVED*  -1
```

```
AutoClass CLASSIFICATION for the 500000 cases
SEARCH SUMMARY 50 tries over  4 hours 51 minutes 33 seconds

PROBABILITY  exp(-6491241.150) N_CLASSES   4 FOUND ON TRY   14 *SAVED*  -1
```

Table 4.7: Autoclass results using only 5 attributes and different dataset sizes.

Up to now, no clear explanation has been found for these seemingly awkward results, although it seems that this version of Autoclass can only manage data and models below a certain combination of dataset size and number of adjusted parameters.

But a possible explication can be related with the fact that Autoclass favours models with small number of classes and simple models. The same than MLE algorithm can finish with a class for each unique case, Autoclass, trying to avoid that behaviour, can finish with a unique class containing all cases.

The fact that the datasets were formed with duplicated instances also has its importance. Final test over datasets without duplicated instances, formed by the original value plus a random value between -0.05 and 0.05 produced results according to the number of classes expected. Thus, Autoclass found 116 classes in a dataset of 150000 instances constructed in this way, and took 1 day 16 hours 58 minutes 35 seconds to process them. When processing 500000 instances in these same conditions, Autoclass could find 168 classes on try 39 and took 7 days 22 hours 23 minutes 43 seconds. (Table 4.8)

AutoClass CLASSIFICATION for the 150000 cases SEARCH SUMMARY 50 tries over 1 day 16 hours 58 minutes 35 seconds PROBABILITY exp(-6049605.560) N_CLASSES 116 FOUND ON TRY 40
AutoClass CLASSIFICATION for the 250000 cases SEARCH SUMMARY 50 tries over 2 days 18 hours 45 minutes 50 seconds PROBABILITY exp(-10106252.800) N_CLASSES 90 FOUND ON TRY 43
AutoClass CLASSIFICATION for the 500000 cases SEARCH SUMMARY 50 tries over 7 days 22 hours 23 minutes 43 seconds PROBABILITY exp(-20048326.500) N_CLASSES 168 FOUND ON TRY 39

Table 4.8: Autoclass results using different dataset sizes with no duplicated instances.

The increase in the number of classes can be explained in the manner in which the new instances were created. Now, a time extrapolation is possible considering that computational time increases linearly with size of the dataset. So, to process 10^8 instances can take with this implementation of Autoclass **4.8 years** what implies the need of speeding up the algorithm by parallelization. This has been already implemented with different strategies in the literature [11] to handle very large data set in reasonable time.

4.4 Sensitivity to new classes

Autoclass is now evaluated on its ability to detect new classes. The experiment consisted in introducing new synthetic data (k instances) in the real dataset with a given mean μ and covariance matrix Σ and checking if Autoclass was able to find it or not.

Autoclass was configured with the following parameters: no random initializations (*force_new_search_p = true, start_fn_type = "block", randomize_random_p = false*), a fixed number of iterations (*max_n_tries = 50*), starting with different number of clusters (*start_j_list = 20,30,40,50,60*), and all attributes following a multivariate normal distribution (*multi_normal_cn 0 1 2 3 4 5 6 7 8 9 10 11 12*).

Table 4.9 shows how the data were generated and the results thus obtained. Data of experiments 7, 8, 9 were generated with mean values in the middle of each attribute range. Data of experiments 10 and 11 were generated in the vicinity of cluster 19. The mean value of each attribute was computed as the mean value plus the standard deviation of that attribute in cluster 19. Synthetic data in experiment 12 were also generated near cluster 19, but the mean values were computed as the mean value plus 0.5 times the standard deviation of each attribute. Figure 4.1 shows the location of synthetic instances in experiments 10 and 11.

Exper.	Parameters to generate synthetic instances	Result
1	k=20, μ=0, Σ=(diagonal 1, rest of values 0)	Mixed
2	k=20, μ=0, Σ =(diagonal 1, rest of values 0.5)	Mixed
3	k=20, μ=0, Σ =(diagonal 1, rest of values 0.9)	Found
4	k=20, μ=-1, Σ =(diagonal 1, rest of values 0.1)	Found
5	k=20, μ=-1, Σ =(diagonal 1, rest of values 0.5)	Mixed
6	k=20, μ=-1, Σ =(diagonal 1, rest of values 0.9)	Mixed
7	k=20 μ={-1,-1,-1.21,-2.5,-2.75,-1.5,-1.5,-2.25,-0.5,-0.5,0.5,1.25,1} Σ=(diagonal 1, rest of values 0.1)	Mixed
8	k=20 μ={-1,-1,-1.21,-2.5,-2.75,-1.5,-1.5,-2.25,-0.5,-0.5,0.5,1.25,1} Σ=(diagonal 1, rest of values 0.5)	Mixed
9	k=20 μ={-1,-1,-1.21,-2.5,-2.75,-1.5,-1.5,-2.25,-0.5,-0.5,0.5,1.25,1} Σ=(diagonal 1, rest of values 0.9)	Found
10	k=50 μ={-0.236,0.279,-0.797,-1.404,-1.869,-2.198,-1.884,-2.441,-0.553,2.09,0.0739,0.654,0.649 } Σ=(diagonal 0.1, rest of values 0.01)	Mixed
11	k=50 μ={-0.236,0.279,-0.797,-1.404,-1.869,-2.198,-1.884,-2.441,-0.553,2.09,0.0739,0.654,0.649 } Σ=(diagonal 0.1, rest of values 0.09)	Mixed
12	k=50 μ={-0.37,-0.116,-0.885,-1.597,-2.1395,-2.459,-2.017,-2.6405,-0.686,1.845,0.04885,0.577,0.585} Σ=(diagonal 0.01, rest of values 0.009)	Mixed

Table 4.9: Parameters used to generate the synthetic clusters and whether Autoclass was able to single it out or mixed it with original data.

More in detail, the results were

1. 85% of the synthetic instances were mainly found in two clusters: Cluster 47 with 12 synthetic instances and 4 real ones and Cluster 56 with 5 synthetic instances and 2 real ones.
2. 95% of the new data were mainly found in two clusters: Cluster 50 with 15 synthetic instances (100%) and cluster 51 with 4 synthetic instances (40%) and 6 real ones.
3. The new data were grouped in cluster 54 with a composition 100% of synthetic instances.
4. The new data were grouped in cluster 48 with a composition 100% of synthetic instances.
5. The new data were split in two clusters: cluster 55 with 7 synthetic instances (100%) and cluster 46 with 13 synthetic instances out of 19 (68%).
6. The new data were grouped in cluster 50 with a composition of 20 synthetic instances and 1 real data.
7. The new data were assigned to Cluster 43 (20 synthetic instances and 2 real ones).

8. The new data were assigned to two clusters: Cluster 50 with 19 synthetic instances and 1 real instance and cluster 66 with 1 synthetic instance and 10 real ones.

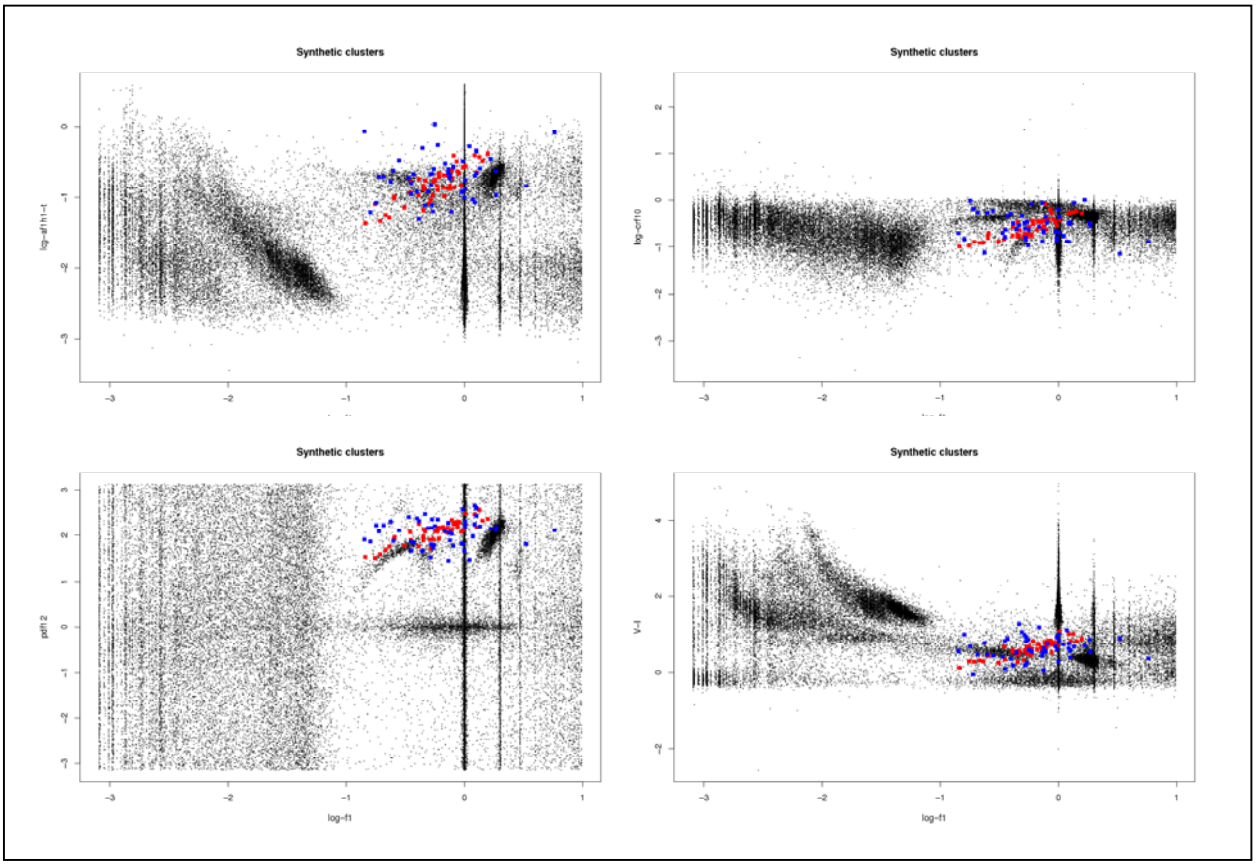


Figure 4.1 - Plot of the location of synthetic instances in experiments 10 (red) and 11 (blue)

The results show that Autoclass is very sensible to detect new classes that follow a multivariate distribution even in the vicinity of existing peaks in the point density distribution. The number of instances used (k=20, k=50) was very reduced and Autoclass showed very good performance detecting them. Even when the result column in Table 4.9 states 'Mixed', i) it is always expected to find real cases close to the synthetic ones, and ii) the percentage of true variables is always below 20%.

4.5 Astrophysical interpretation

We tried a cluster interpretation taking the results obtained on trial 6h processing (*max_duration = 21600*), without random initializations, (*force_new_search_p = true, start_fn_type = "block", randomize_random_p = false*) starting with list (*start_j_list = 20,30,40,50,60*), and all atributes following a multivariate normal distribution (*multi_normal_cn 0 1 2 3 4 5 6 7 8 9 10 11 12*) and applied over the total dataset (13 attributes, 43351 cases).

We also used the OGLE test datasets belonging to the classical variable stars. We asked Autoclass to predict class membership of these previously labeled examples. Table 4.10 shows the results of prediction in the form of percentages of class assignments, that is, what percentage of the original dataset was assigned to cluster number i. We only show results with a significance above than 5%. The main conclusions based on the visual analysis of two-dimensional plots are summarized in the following sections.

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
4							6.03	22.74		
5										81.23
7			28.57		6.79	21.25	78.62			
8								27.89		
14								18.02		
15								13.89		
16				35.79	30.24					
17				30.11	24.69					
18				7.9	14.19					
19	49.35		42.85							
20				16.17	16.04	65				
21	39.83									
23										10.47
24		92.95	21.42						94	6.52
28								6.98		
29			7.14							

Table 4.10: Autoclass prediction of class membership for labeled instances.

Clusters 0 and 6: Ogle Small Amplitude Red Giants or OSARGS

Cluster 0, the largest with 5199 instances, corresponds to the so called OGLE small amplitude. In order to prove this assertion we include two plots of the examples in cluster 0 representing the V −I colour as a function the logarithm of the first (Figure 4.2) and second (Figure 4.3) detected frequencies.

In Figure 4.2 we have included three labels A, B and D with the proposed location of the corresponding sequences described in Soszynski et al. (2004).

According to Soszynski et al., objects in the A sequence (OSARGS) have secondary frequencies in the A, B and D sequences (see Figure 4.2 of their work) with diffuse and negligible contribution to the C and C" sequences. We see in Figure 4.3 that the second frequency of our cluster 0 behaves as described by Soszynski et al. if our sequence identification is correct. As a reinforcement to this identification, it must be stated that (linear) correlations between V−I colour and first and secondary periods agree both quantitative and qualitatively with the values in Soszynski et al. (2004).

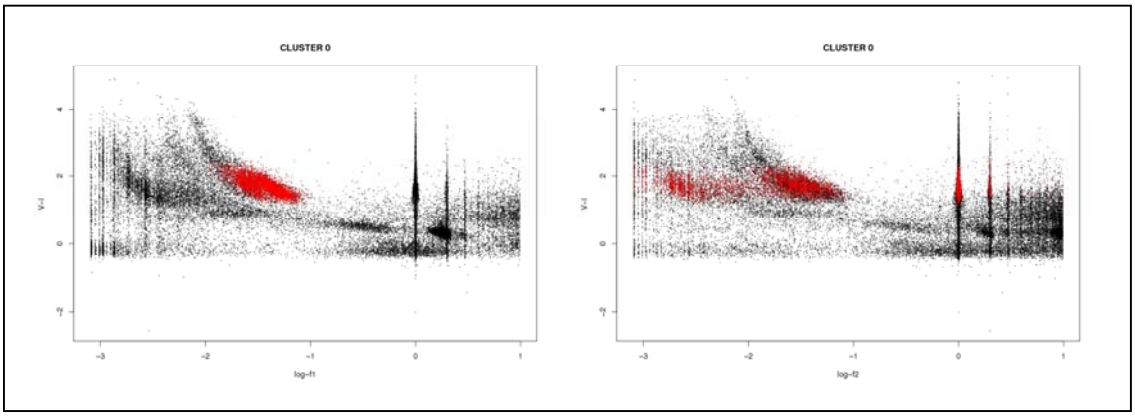


Figure 4.2 and 4.3: Plots of the location of instances in cluster 0 in the log-f1 (logarithm of the first frequency)- (V − I) colour index plane and in the log-f2 - (V-I).

Interestingly, cluster 6 with 1878 instances also represents OSARGS but with inverted ordering of frequencies, that is, with a first frequency in the D sequence and the second frequency in the A or B sequences. (Figures 4.4 and 4.5)

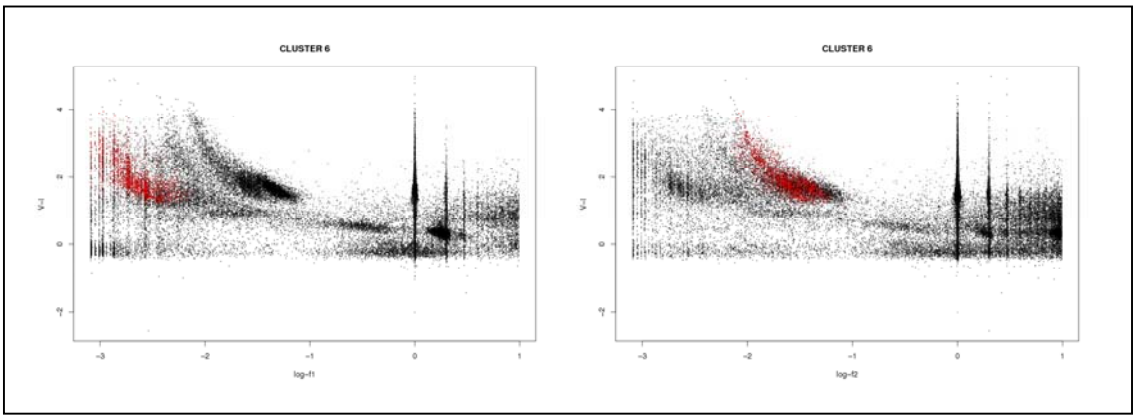
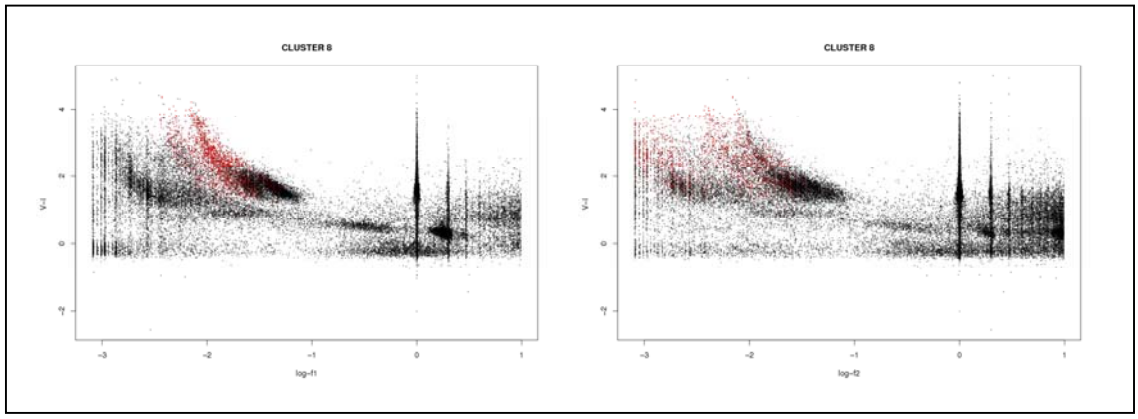


Figure 4.4 and 4.5: Plots of the location of instances in cluster 6 in the log-f1 (logarithm of the first frequency)- (V − I) colour index plane and in the log-f2 - (V-I).

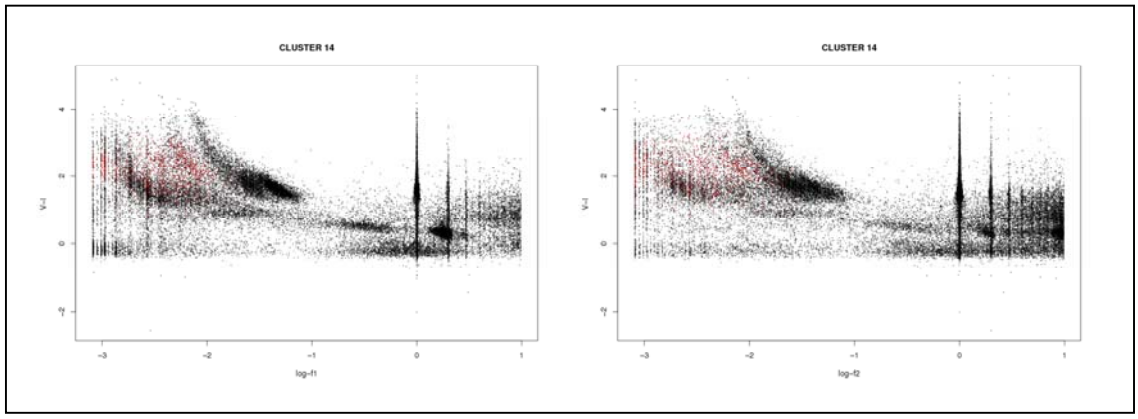
Clusters 8, 14 and 28: Mira and semirregular stars

Sequences C and C" are less visible in the survey plots. They should appear between sequences A, B and D, and group Mira and Semirregular variables. These sequences (C and C"), if correctly identified, must have secondary

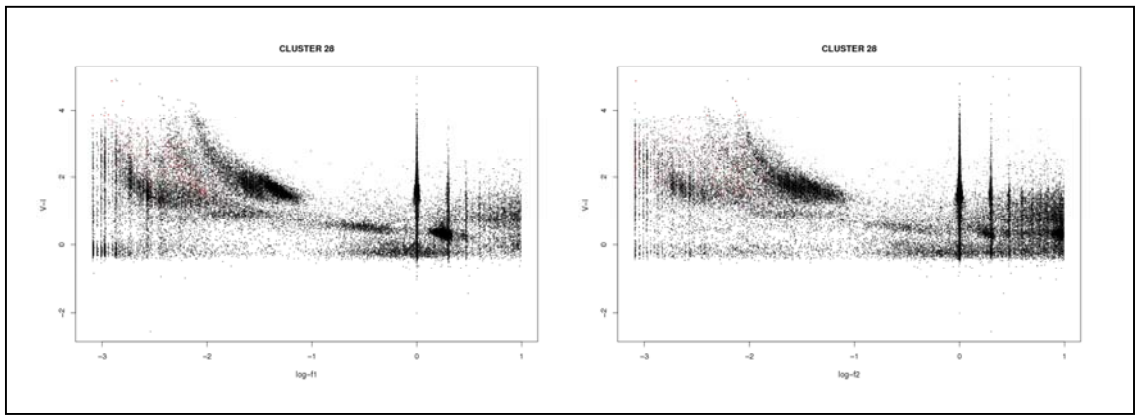
frequencies in the C, C" and D sequences as is actually the case (see Figures 4.6-4.7, 4.8-4.9 and 4.10-4.11), therefore confirming our initial identification.



Figures 4.6 - 4.7: Plot of the location of instances in cluster 8 in the log-f1 - (V−I) and log-f2-(V-I)



Figures 4.8-4.9: Plot of the location of instances in cluster 14 in the log-f1 - (V−I) and log-f2-(V-I)



Figures 4.10-4.11: Plot of the location of instances in cluster 28 in the log-f1 - (V−I) and log-f2 - (V-I)

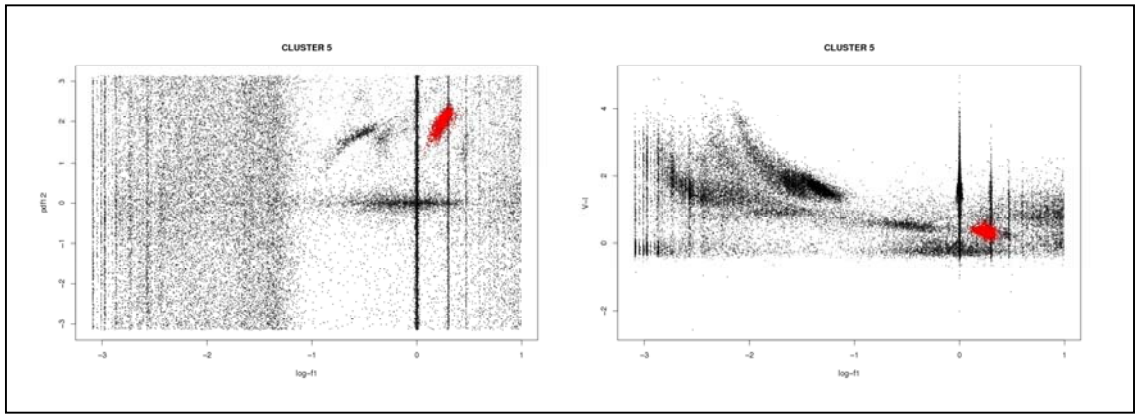
Clusters 5, 23 and 24: RR Lyrae stars

Clusters 5 and 23 are basically defining the same locus on the multidimensional space of parameters (see Figures 4.12-4.13 and 4.14-4.15). The only apparent

difference in the 2D projections is a larger scatter of the colour indices in the smaller cluster (cluster 23, 418 instances) with respect to the main one.

Cluster 24 is composed of variable stars with slightly higher frequencies than the other two clusters (see Figures 4.16-4.17) and lower amplitude ratios of the first two harmonics of the most significant frequencies R_{21}. They occupy the locus of the RRc and RRd (double mode RR Lyrae) as confirmed by the log-f1 vs. log-f2 plots. Therefore, we interpret this cluster as grouping the more sinusoidal light curves of RRc stars and double mode RR Lyrae pulsators. There is a separatrix between the two subcomponents according to their frequency ratios that is not found significant enough by Autoclass so as to separate the two groups. This is because the two subcomponents seem to share common ranges for all other parameters, although this separation sensitivity would be desirable.

Comparison plots created solely with preclassified stars can be found in Soszynski et al. (2003) and Sarro et al. (2008).



Figures 4.12-4.13: Plots of the location of instances in cluster 5 in the log-f1 -φ_{21} plane, where φ_{21} is the pdf12 attribute listed in Table 3.1 and in the log-f1 - (V−I) colour index plane.

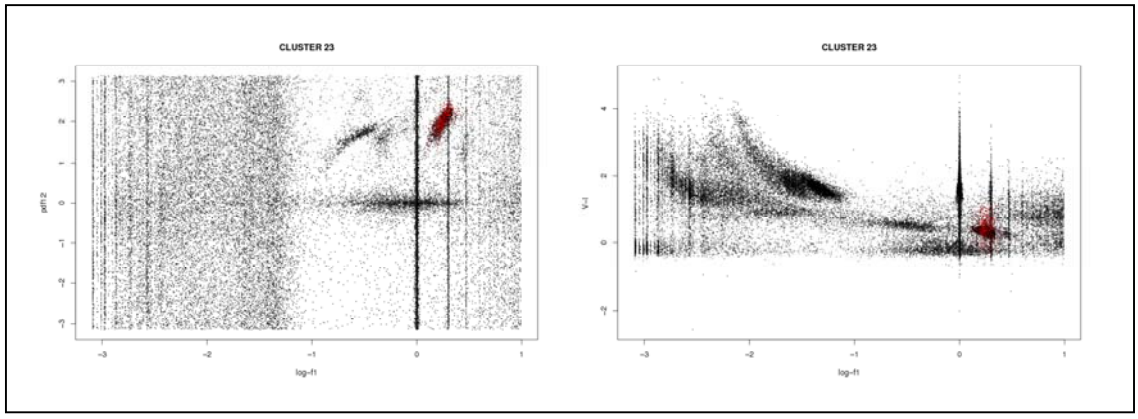
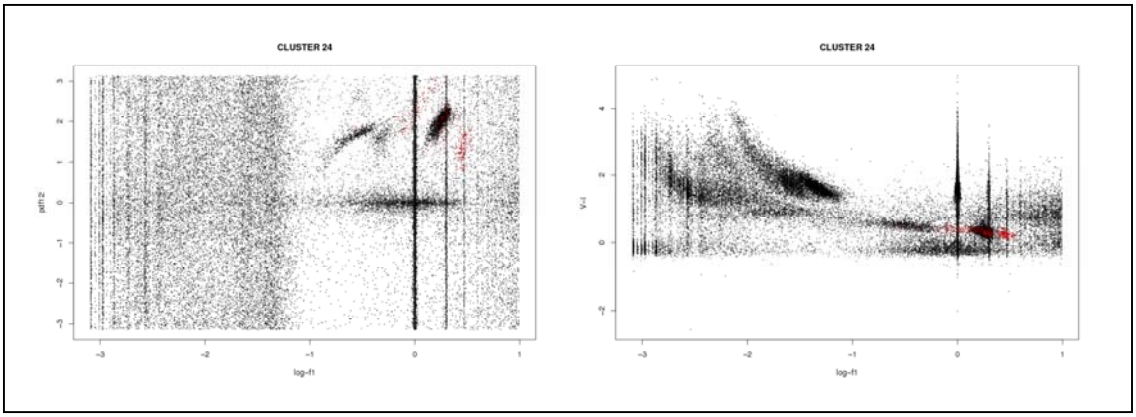


Figure 4.14-4.15: Plots of the location of instances in cluster 23 in the log-f1 -φ_{21} plane, where φ_{21} is the pdf12 attribute listed in Table 3.1 and in the log-f1 - (V−I) colour index plane.

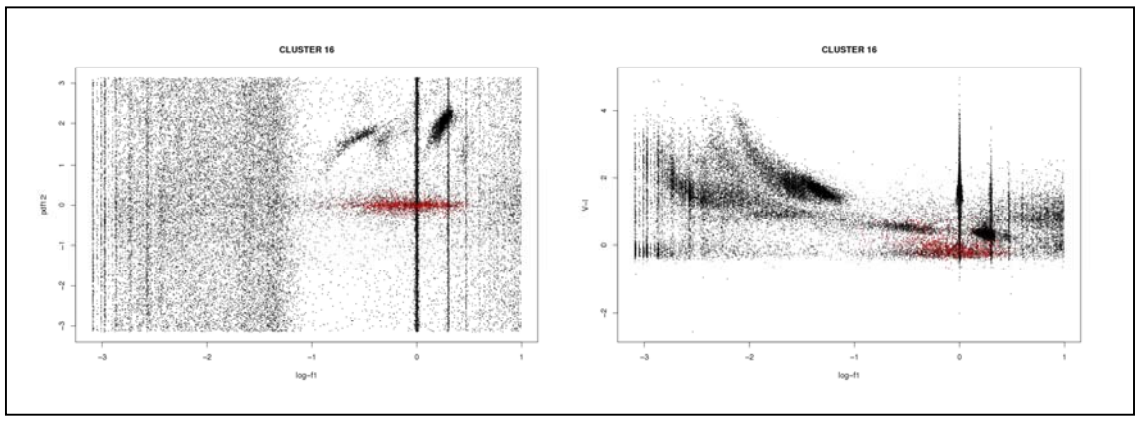


Figures 4.16-4.17: Plots of the location of instances in cluster 24 in the log-f1 -φ_{21} plane, where φ_{21} is the pdf12 attribute listed in Table 3.1 and in the log-f1 - (V−I) colour index plane.

Clusters 16, 17, 18 and 20: eclipsing binary systems

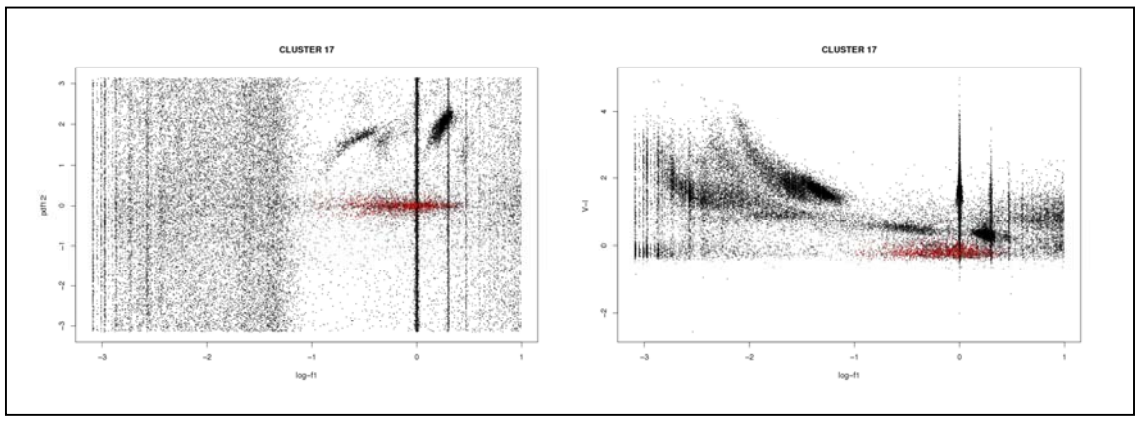
Clusters 16 and 17 represent the same type of stars (eclipsing binaries) the only difference being the value of the second frequency. In all instances of both clusters, the second frequency is a spurious detection with a characteristic ratio for each cluster. Whereas clusters 16 and 17 are clearly defined by a vanishing value of the phase difference between harmonics of the first frequency (φ_{12}) and low values of the colour indices (see figures 4.18-4.19), we see that cluster 18 contains some contamination of spurious detections (log-f1 = 0.0) with random values of φ_{21} (and colour indices consistent with those of clusters 16 and 17), and cluster 20 is characterized by the same vanishing values of φ_{21} but with a larger scatter of colour indices and first frequencies.

Most interesting, there is a clear correlation between the amplitudes of the increasing harmonics and the cluster, in the sense that cluster 16 groups eclipsing binaries with the largest departures from sinusoidality (most detached systems, with narrow eclipses and high values of harmonic amplitudes) and subsequent clusters are characterized by decreasing values of the higher harmonics and, therefore, more sinusoidal light curves characteristic of semidetached and close binary systems. This clustering has to be investigated further in order to check if it supports the new classification scheme by Sarro et al. (2006) or, on the contrary, is more in agreement with the traditional EA, EB, EW classification system.



Figures 4.18-4.19: Plot of the location of instances in cluster 16 in the log-f1 (logarithm of the first frequency)- φ12 plane and in log-f1 - (V-I) color index plane.

.



Figures 4.20-4.21: Plot of the location of instances in cluster 17 in the log-f1 (logarithm of the first frequency)- φ12 plane and in log-f1 - (V-I) color index plane.

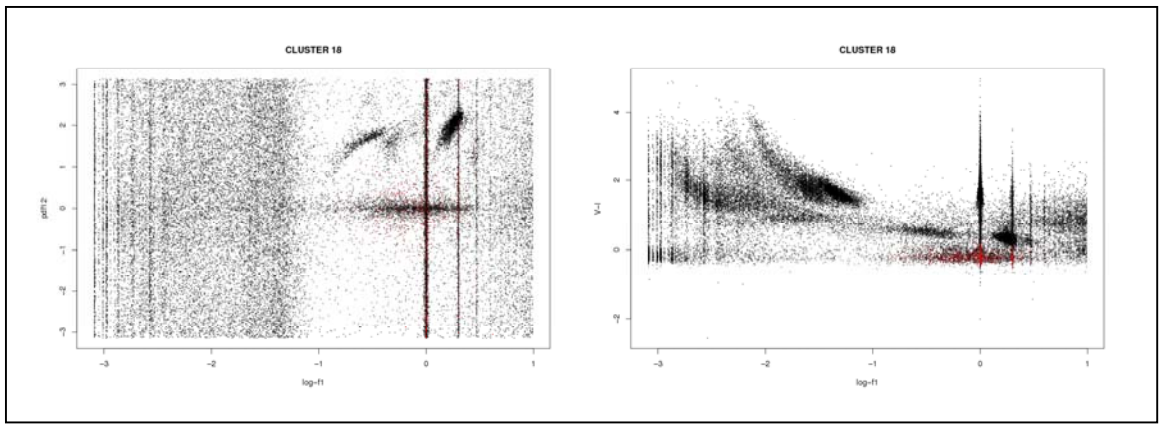
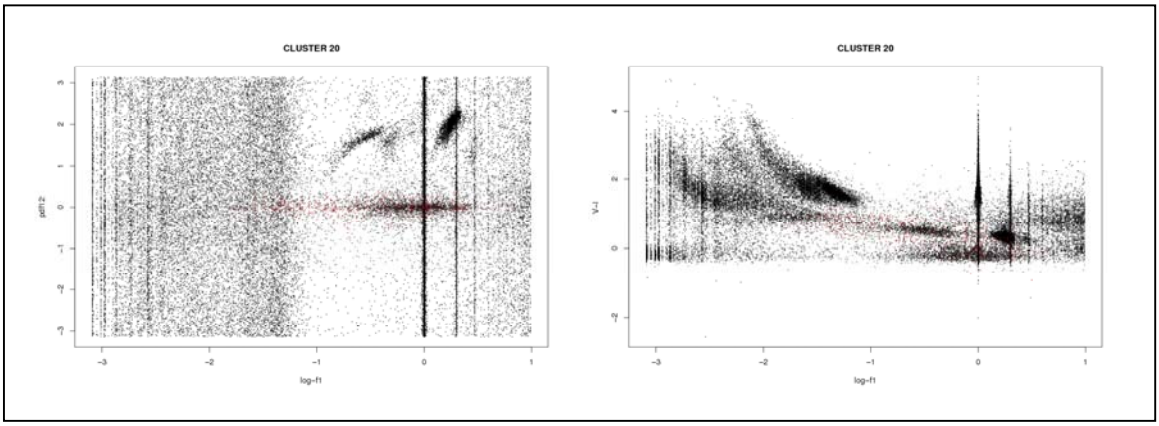


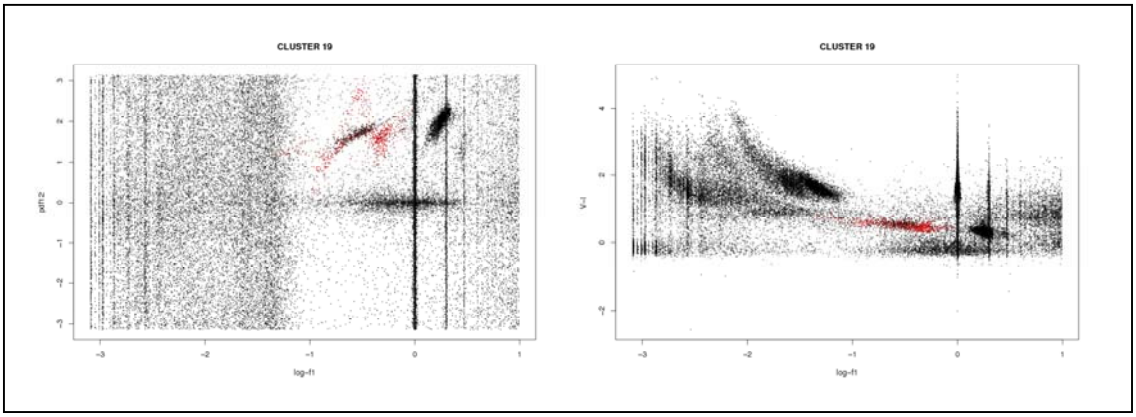
Figure 4.22-4.23: Plot of the location of instances in cluster 18 in the log-f1 (logarithm of the first frequency)- φ12 plane and in log-f1 - (V-I) color index plane.



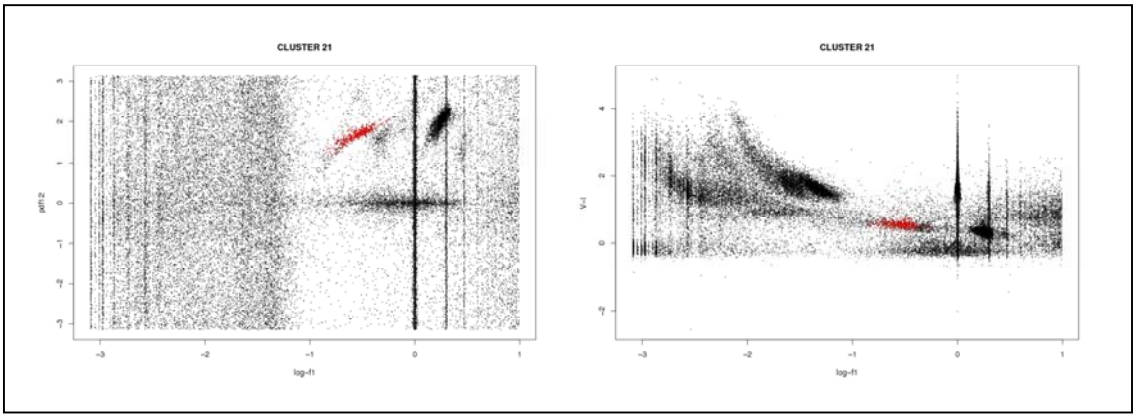
Figures 4.24-4.25: Plot of the location of instances in cluster 20 in the log-f1 (logarithm of the first frequency)- φ12 plane and in log-f1 - (V-I) color index plane.

Clusters 19 and 21: Cepheid stars

The most meaningful tools for the interpretation of neighboring clusters 19 and 21 are the log-f1 vs (V − I) and φ_{21} (the phase difference between the first two harmonic components of the first frequency). Figures 4.26-4.27 show two such plots for cluster 19 and Figures 4.28-4.29 for cluster 21. It is evident from the plots that Autoclass is separating first overtone pulsators (cluster 19) and fundamental mode cepheids (cluster 21). For a confirmation of this assertion, equivalent plots in Udalski et al. (1999) constructed only with preclassified cepheids in the LMC can be consulted.



Figures 4.26-4.27: Plot of the location of instances in cluster 19 in the log-f1 (logarithm of the first frequency)- φ_{21} plane.



Cluster 7 : ellipsoidal stars

While previous clusters were clearly identified on the basis of their properties, Cluster 7 has only been identified with the aid of examples labeled by the OGLE team. Among the several samples of variability types in the LMC, SMC and the bulge, the sample of ellipsoidals in the LMC was used to interpret the clusters (see below). It turns out that approximately 80% of stars in the sample of LMC ellipsoidals were classified in cluster 7. Although they do not seem to form a particular subset of cluster 7, one can not interpret that all instances in cluster 7 (1871) are ellipsoidals due to the large ranges spanned in frequencies, amplitudes and amplitude ratios. Only in the log-f1 vs (V − I) space (Figure 4.30) does the cluster appear sharply defined.

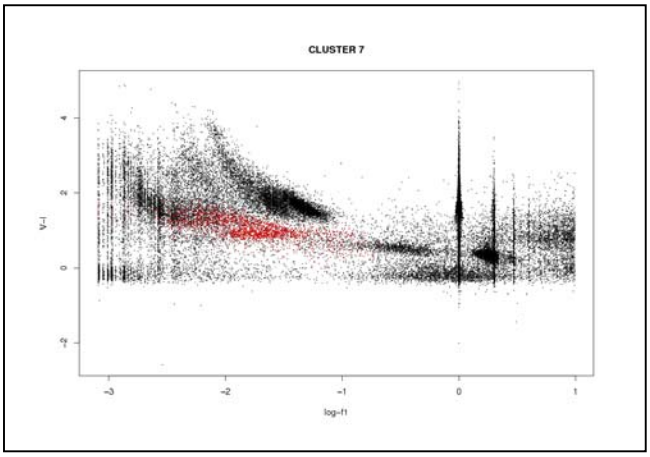


Figure 4.30: Plot of the location of instances in cluster 7 in the log-f1 (logarithm of the first frequency)- (V − I) colour index plane.

Cluster 13 : BE stars

Cluster 13 occupies the locus characterized by short frequencies and blue colour indices (see Figure 4.31). In order to interpret the astrophysical content

of this cluster we have investigated the classification of objects in the Hipparcos database falling in this same region of the parameter space. At least for the bluest objects of the cluster with the lowest frequencies, there was a remarkable prevalence of BE variables according to the SIMBAD database.

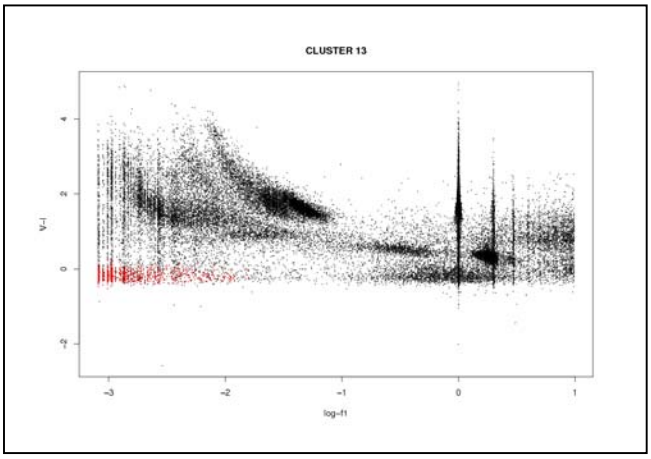


Figure 4.31: Plot of the location of instances in cluster 13 in the log-f1 (logarithm of the first frequency)- (V − I) colour index plane.

Multiperiodic variables

In this section we used other datasets to try to identify clusters of multiperiodic variables stars. Obviously, the OGLE database is not necessarily well suited for these target classes, so it comes as no surprise that the results are not satisfactory. The input data were taken from the classification carried out by the hierarchical classifier presented in Sarro et al. (2008), and include β Cephei stars (292 cases), δ Scuti stars (22), γ Doradus (102), Pulsating Variable Super Giants (PVSG; 79), and Slowly Pulsating B stars (590). We used Autoclass to predict class membership of these examples, some of which are indeed spurious frequency detection at the alias frequencies of the OGLE LMC survey. The results can be found on Table 4.11 and, again, the figures shows percentages above 5%. It is evident that the largest fraction of candidates (we would like to stress that these are mere candidates) falls in cluster 1 that groups the shortest period variables (most stars with periods of several cycles per day and outside the classical RR Lyrae locus are in this cluster). All other significant contributions to the table are compatible with the clusters assigned and the contamination expected in the candidate lists.

Cluster	All-2-bcep-8	All-2-dscut-8	All-2-gdor-8	All-2-pvsg-7	All-2-spb-8
1	25.68	75.94	75.49	68.64	13.63
3				7.62	
9	6.5			9.49	
10					9.09
11	9.58	7.59	22.54	5.42	9.09
12		5.06			
13	30.47				
18	12.67				45.45
22	14.04				13.63
29		5.06			

Table 4.11: Autoclass prediction of class membership for multiperiodic data instances

4.6 Experiments with a reduce attribute set.

Our next experiment was to run Autoclass to find a solution using only 5 out of the original 13 attributes (log-f1, log-af1h1-t, log-crf10, pdf12, V-I), and predict class assignments for the same set of labeled examples taken from the OGLE database. We tried a new cluster interpretation taking the results obtained on 6h processing (*max duration = 21600*), without random initializations, (*force_new_search_p = true, start_fn_type = "block", randomize_random_p = false*) starting with list (*start_j_list = 20,30,40,50,60*), and using the 5 attributes following a multivariate normal distribution (*multi_normal_cn 0 2 8 9 12*) and ignoring the rest, and applied over the total dataset (43351 cases).

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
0								15.24		
2										80.76
5			14.28							
6								22.92		
8				32.02	40.12					
10				33.6	14.81					
11							7.34			
13					5.55					
15									20	
20				18.72	17.28	13.75				
21								6.14		
25								16.52		
27						13.75				
28	42.49									
29										12.78
30								5.3		
31							6.85			
32								8.7		

33						7.5			
35					7.5	50.57			
38	13.17	21.12	21.42					76	
40	26.19	69.01	7.14						
45					17.5				
48		5.63							
49			14.28						
52							8.55		
55			7.14		13.75				
62			14.28						
67	5.55		21.42		13.75				

Table 4.12: Autoclass prediction of class membership for labeled instances using only 5 attributes

Under these setup, the class predictions were distributed according to the results shown in Table 4.12. This time Autoclass found 73 clusters. Again, we only show results when the percentage of instances in a given cluster is above 5%. With 5 attributes, the vectors of probabilities become more diffuse thus making the cluster assignments less clear.

The increasing number of clusters (52% more) with respect to the model with 13 attributes could be interpreted as an overfitting to the data, easier with lesser attributes and models with many parameters as the multivariate gaussian.

The results with 5 attributes instances can be summarizes as follows:

1. Long period variables continue being the best separated class, with little contamination from other classes.
2. RR Lyr stars are still well separated, but double mode RR Lyrae stars (RRd) are now clustered together with Cepheids. This effect was also encountered in Sarro et al. (2008) and deserves further investigation.
3. Eclipsing binaries keep being well resolved.
4. Fundamental and First Overtone Cepheids and Double Mode Cepheids that were separated on the basis of second frequency information, are now clustered, as expected, together.
5. Ell-ecl and ell-ell are now separated.

In general, this trial shows less resolving power at separating clusters as a consequence of the reduce attribute space.

4.7 Autoclass applied to cluster the database of labeled examples.

Finally, we used Autoclass to find a solution over the sum of all test datasets (10063 instances). These datasets are supposed to gather classical variable stars characteristics (avoiding low signal-to-noise detections and spurious detections) so the clustering has to be simpler. The processing conditions were

the same as in the previous experiment (see section 4.5) but using again the original set of 13 attributes. Autoclass found 35 classes and the analysis of the results (Table 4.13) implies that the classification is very similar to the one obtained on the OGLE LMC dataset: LPVs (Long Period Variables) and RR Lyrae clusters well separated, RRd stars are again well separated, and the rest of types with similar entries in the contingency table.

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
0										62
1				34.29	28.39					
2								31.44		
3								22.88		
4				20.1	10.49					
5						8.75	79.93			
6	37.92									
7								16.16		
8	28.25	67.6	64.28							
9								14.44		
10				13.37	11.11	56.25				
11				12.16	20.37					
12										13.25
13	19.04									
14		28.16	7.14			8.75	7.34			
15								6.21		
16										6.05
17										5.94
18									94	
19			28.57			11.25				
20					6.79					
21					7.4					
24					5.55	6.25				
25	7.69									

Table 4.13: Clustering structure using only labeled datasets. Only percentages above 5% are shown for clarity.

4.8 Effect of considering log-normal attributes

Our previous experiments were performed under the assumption that the attributes log-f1, logf2, log-af1h1-t, log-af1h2-t, log-af1h3-t, log-af1h4-t, log-af2h1-t, log-af2h2-t, log-crf10 were normally distributed, so f1, f2, af1h1-t, af1h2-t, af1h3-t, af1h4-t, af2h1-t, af2h2-t, crf10 were log-normally distributed.

In order to verify this hypothesis, we repeated the experiment using a linear scale for attributes 1, 2, 3, 4, 5, 6, 7, 8, 9. Autoclass was configured with the following parameters: different number of clusters (*start_j_list = 20,30,40,50,60*), 6 hours of calculi (*max_duration = 21600*) and all attributes

without the log-transformation following a multivariate normal distribution (*multi_normal_cn 0 1 2 3 4 5 6 7 8 9 10 11 12*). In 6 hours, Autoclass was only able to perform 11 iterations (in comparison to the 43 performed previously) because it needs more cycles to satisfy the convergence criterion (this criterion was not relaxed with respect to the previous tests). The parameter *max_cycles* was fixed to 1000 and even with this value, most of the tries were non-convergent. Under this conditions, Autoclass found 51 classes.

We also applied the anti-log transformation to the test datasets to validate the results of clustering. The results of Autoclass prediction of class membership are shown in Table 4.14.

The membership pattern is similar to the one obtained considering log-normal variables.

Cluster	cep	dmcep	ptcep	ecl	new_ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
1								42.23		
2			14.28		9.87	40	39.47			
5						7.5	15	8.33		
6										55.94
9								8.26		
13				39.6	32.71	25				
16								6.69		
18	65.49		7.14							
19								11.11		
20										24.04
21								5.48		
23		5.63	7.14						6	
24				11.34	9.87					
27							5.05			
28				13.53	12.34					
29		21.12	21.42			10				
33									68	13.09
34				13.57	9.87					
35				6.48	11.11					
36	6.47	73.23	21.42						26	5.12
40	18.58		21.42							
45			7.14				26.1			

Table 4.14: Autoclass predictions with antilog attributes.

This could be interpreted as, although the total data points form a skewed distribution, the density of points that forms clusters do not so in that way. Considering that a log-normal and a normal distribution are almost visually indistinguishable if sigma values are low, this is what is happening in most clusters. Outliers points make the main difference between the two probability models.

4.9 Comparison with Hipparcos dataset

In this section, we apply Autoclass to the Hipparcos dataset (2498 instances) with exactly the same attribute information as the original OGLE dataset.

First of all, we used Autoclass to find the best solution after 6 hours of processing (*max_duration=21600*), without random initialization, (*force_new_search_p = true, start_fn_type = "block", randomize_random_p = false*), starting with different number of clusters (*start_j_list = 20,30,40,50,60*), and all attributes following a multivariate normal distribution (*multi_normal_cn 0 1 2 3 4 5 6 7 8 9 10 11 12*).

Autoclass best solution is composed of 13 clusters, but in this experiment most of the best models are actually duplicates, that is, solutions found already several times in the six hours run. Not only is the best solution composed of a smaller number of clusters, also the variance of the number of clusters found in the 10 best solutions is much smaller than when calculated for the OGLE runs.

The class membership prediction for the classical variable stars datasets are shown in Table 4.15. It is evident that Autoclass finds clusters with less astrophysical homogeneity in this much smaller dataset, as expected given the seemingly more diffuse Probability Density Function (PDF) of the Hipparcos dataset (see Figure 3.2).

Cluster	cep	dmcep	ptcep	ecl	new_ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
0							9.95	44.86		
1	6.47	25.35	21.42	51.11	58.64	83.75	85.31	18.86	56	31.58
2		52.11			6.79				36	
3	79.05	14.08	78.57							12.54
4	5.78			37.21	27.16					40.38
5								5.15		
8								20.51		
11		5.63		5.14	5.55					12
12									6	

Table 4.15: Autoclass prediction of class membership for OGLE labeled instances using the clustering structure inferred from the Hipparcos dataset.

We also tried to match these cluster with the ones obtained with the OGLE LMC dataset. In order to perform this match, we used Autoclass to produce OGLE clusters assignment probabilities for each set of instances defining a Hipparcos cluster. The results are shown in Table 4.16. Rows correspond to OGLE clusters, and columns, to Hipparcos clusters. Cells contain the percentage Hipparcos instances that are members of each OGLE cluster. This table shows that there is not a clear correspondence among clusters of both datasets.

Cluster	0	1	2	3	4	5	6	7	8	9	10	11	12
4							15.06	8.69					
5					9.14								
7		29.48		11.51								28.57	25
8								8.69					
13		10.62											
14							9.58		27.27				
16					15.33								
17					17.1								
18			11.55										
19				31.74									
20		10.62		14.88	43.95						12.5		
22			38.94									14.28	
23					5.6								
24			5.52	5.33									
25		10.07				18.89							
28						11.81	12.32						
29			27.63	16.57								35.71	
31	21.25					45.66	30.13	34.78	13.63	15	12.5		
32							5.47						
34	17.32	12.08	10.3	14.32						10	6.25	21.42	
37	6.6									10			
38											6.25		
40	18.75	5.67				9.44	12.32	8.69	27.27	30	43.75		
43									13.63				
44	16.07					9.44	6.84	30.43		15	18.75		25
46													50

Table 4.16: Contingency table for clusters found in the OGLE and Hipparcos datasets.

5. Results of application of HMAC

In this section, we present the results obtained with HMAC applying the same evaluation criteria than the specified ones for Autoclass. Any of them, as the impact of randomness, is not applicable because of the irrelevance of initialization of this algorithm but this criterion and others are maintained by coherence.

For Autoclass, we assumed that attributes followed a multi normal distribution. But, in this implementation of HMAC this hypothesis will not be important because of this approach can be used to find modes of any density in the form of a mixture distribution.

5.1 Impact of randomness in clustering results.

Nonparametric clustering approaches present among their advantages the irrelevance of initialization. To test this fact, HMAC algorithm was run over the OGLE Large Magellanic Cloud dataset with 43351 cases of variable stars ordered in two different ways to verify that the results remained the same.

HMAC(mtree) was configured to perform a hierarchical clustering over the total dataset and using 13 attributes. The rest of parameters were configured with its default values (*step size of the bandwidth sequence=0.1, maximum bandwidth in sequence = 2.0*). This invocation of HMAC search took 1 day 22 hours 56 minutes and 5 seconds and gave a dendrogram of 8 levels shown in Table 5.1:

Level	1	2	3	4	5	6	7	8
Band-width	0.1694400	0.3388799	0.5083199	0.6777598	0.8471998	1.016640	1.186080	1.524960
# clusters	36289	4306	260	32	5	3	2	1
Size 1st cluster	2463	3599	8119	14545	23363	30609	30609	43351
Size 2nd cluster	521	2977	5063	8132	12740	12741	12742	
Size 3rd cluster	341	2129	4956	7244	7246	1		
Size 4rd cluster	315	2036	4287	6455	1			
Size 5rd cluster	225	1995	4277	5899	1			
Size 6rd cluster	209	1825	2892	337				

Table 5.1. Dendrogram obtained by HMAC search over the OGLE LMC dataset

The second run over the disordered dataset gave exactly the same results what confirms the hypothesis of the irrelevance of initialization.

5.2. Impact of computation time on clustering results

The next criterion to evaluate HMAC is the impact of computation time on clustering results. This is other criterion without sense in the context of HMAC, because the computation time needed to find a solution in a fixed dataset is always the same and the solution that the algorithm finds too.

From the results of section 5.1 we could see that the hierarchical clustering can be an intensive processing task. We also saw that, although level 3 has 260 clusters, there are only 54 clusters with more than 5 instances. We considered these 206 clusters no relevant. The same happens with level 4, with 32 clusters, only 7 have more than 5 instances. So, we focused on the results that could be obtained with bandwidth (sigma) values between 0.50832 and 0.67776.

The objective of the experiment was to quantify if processing time could be reduced using these preliminary results and focusing us in these bandwidth values avoiding preliminary steps of hierarchical clustering.

Again, HMAC(mtree) was configured to perform a hierarchical clustering over the total dataset and using the 13 attributes. We executed the algorithm with different sigma range values. Table 5.2 shows these results.

Run	Bandwidth values (clusters)	Processing time (sec)
1	0.63	1d 11h 1m 45s
2	0.61(72), 0.63(58), 0.65(39)	1d 15h 23m 30s
3	0.60(89), 0.63(58), 0.66(35)	1d 8h 9m 36s = 115776s

Table 5.2. Sigma values and processing time needed to test them.

The number of clusters obtained was coherent, but not equal, with the sigma value independently from the starting point of the search. So, to have a preliminary general clustering to quantify bandwidth values of interest can be helpful to reduce processing time.

Another question we wanted to answer is if a hierarchical clustering is needed. So we repeated run 3 in the same conditions as before but without enforcing nested hierarchy. This invocation of HMAC-search(mtree) took 4 days 6 hours 49 minutes 45 seconds, so to enforce nested hierarchy is needed if we want to test several sigma values and reduce processing time. The clusters obtained in the first hierarchical level were equal, but not in the next levels. The results are logical due to the reduction of data to process for consecutive levels in nested hierarchy.

However, the most important factor in time reduction comes from a reduction in data dimension as it is explained in the next section.

5.3. Impact of dataset size on computation time.

The next investigated aspect is the increment of processing time with increasing dataset size, assuming that the algorithm must be able to handle the order of 10^8 instances.

In order to perform our experiments we took the original dataset of 43351 cases and we duplicated instances to get datasets of 100000, 200000, 500000 and 1000000 cases. HMAC(mtree) was configured to perform over the total dataset and using the 13 attributes. We executed the algorithm with only one sigma value (0.58). The experiment had to be aborted due to the long processing time required (more than a week for 100000 instances). The experiment was then reduced to process datasets of 25000, 50000 and 100000 instances. Table 5.3 shows the results thus obtained and a poor extrapolation of time needed to process 10^8 instances in these conditions.

Dataset size	Processing time (sec)
25000	8 hours 55 min 58 sec = 32158 sec
50000	1 day 22 hours 41 min 34 sec = 168094 sec
100000	7 days 2 hours 23 min 49 sec = 613438 sec
10^8	**791357490.5 sec = 25.09 years**

Table 5.3. Time needed to process 25000, 50000 and 100000 instances with only one sigma value.

These time results are only one iterative step of the hierarchical clustering. Without any code optimization, this algorithm, as it is presented, seems to be non viable of applying in the context of CU7. This time estimation exceeds 500% the Autoclass estimation.

But it is possible to reduce computation time using an hybrid clustering method. A preliminary clustering as k-means can be applied to reduce data dimension. The number of clusters resulting from this preliminary clustering has to be large enough compared with the desired number of clusters to retain the topological structures in the non parametric density estimate. The purpose of this preliminary clustering is more of quantizing than clustering.

We do not show the time reduction that this measure for enhancing speed produces because it also depends on the preliminary clustering method selected, but further, in this report this optimization technique is used with the aim of reducing memory requirements of this algorithms with good results.

A combination of procedures is also shown in section 6, with the aim of improving clustering that also has effect reducing computing time.

5.4. Sensitivity to new classes.

HMAC is now evaluated on its ability to detect new classes. The experiment consisted in introducing new synthetic data (k instances) in the real dataset with a given mean μ and covariance matrix Σ and checking if this approach was able to find it or not.

HMAC(mtree) was configured to perform a hierarchical clustering over the total dataset and using 13 attributes. The rest of parameters was configured with its default values (*step size of the bandwidth sequence=0.1, maximum bandwidth in sequence = 2.0*).

Data of experiments are the same than the used for Autoclass. But the number of trials was reduced due the high processing time that HMAC requires. Table 5.4 shows the experiments carried out and its correspondence with the Autoclass ones.

9	k =20 μ ={-1,-1,-1.21,-2.5,-2.75,-1.5,-1.5,-2.25,-0.5,-0.5,0.5,1.25,1} Σ =(diagonal 1, rest of values 0.9)	Not found
10	k =50 μ ={-0.236,0.279,-0.797,-1.404,-1.869,-2.198,-1.884,-2.441,-0.553,2.09,0.0739,0.654,0.649 } Σ =(diagonal 0.1, rest of values 0.01)	Not found
11	k =50 μ ={-0.236,0.279,-0.797,-1.404,-1.869,-2.198,-1.884,-2.441,-0.553,2.09,0.0739,0.654,0.649 } Σ =(diagonal 0.1, rest of values 0.09)	Not found
12	k =50 μ ={ -0.37,-0.116,-0.885,-1.597,-2.1395,-2.459,-2.017,-2.6405,-0.686,1.845,0.04885,0.577,0.585} Σ =(diagonal 0.01, rest of values 0.009)	Not found
13	k =50 μ ={ -0.37,-0.116,-0.885,-1.597,-2.1395,-2.459,-2.017,-2.6405,-0.686,1.845,0.04885,0.577,0.585} Σ =(diagonal 0.01, rest of values 0)	Mixed

Table 5.4. Parameters used to generate the synthetic clusters and result of HMAC detection.

More in detail, the results were:

9. In level 1 and 2 , each synthetic instance forms a cluster. In level 3, the new data were found in 15 clusters: cluster 2 with 1 synthetic instance and 4956 real ones, cluster 3 with 5 synthetic instances and 332 real ones, the rest are in 4 clusters, three singletons and one with two instances. In level 4, most of the synthetic instances have been absorbed by big clusters: cluster 0 with 8 synthetic instances and 6455 real ones, cluster 1 with 7 synthetic instances and 14545 real ones, cluster 4 with 1 synthetic instances and 7244 real ones, cluster 6 with 2 synthetic instances and 5899 real ones. The two synthetic instances that remains are singletons.

10. Only in level 1 with bandwidth 1.694614e-001, synthetic instances were found not totally mixed with real ones. In this level, the new data were found in 18 clusters: Cluster 32547 with 28 synthetic instances 1 real one, cluster 33445 with 5 synthetic instances and 1 real one, cluster 36288 with 2 synthetic instances, and finally, 15 cluster with just 1 synthetic instance). In level 2, bandwidth 3.389228e-001, all the synthetic instances are diluted in clusters with real instances: cluster 20 with 34 synthetic instances and 3633 real ones, cluster 27 with 4 synthetic instances and 731 real ones and cluster 49 with 12 synthetic instances and 480 real ones.

11. In level 1 with bandwidth 1.694614e-001, each synthetic instance is a singleton. In level 2, with bandwidth 3.389237e-001, the synthetic instances are totally mixed with real instances: cluster 20 with 26 synthetic instances and 3600 real ones, cluster 27 with 16 synthetic instances and 728 real ones, cluster 49 with 6 synthetic instances and 483 real ones and clusters 4269 and 4270 with just a synthetic instance.

12. In level 1 with bandwidth 1.694339e-001, the 50 synthetic instances are found in cluster 1104 together with 47 real instances. In level 2 with bandwidth 3.388679e-001, cluster 90 is formed by 50 synthetic instances and 189 real ones. In level 3, with bandwidth 5.083018e-001, cluster 11 has the 50 synthetic instances and 5573 real ones.

13. In level 1, with bandwidth 1.694350e-001, the 50 synthetic instances are found in cluster 5849 together 39 real instances. In level 2, cluster 90 is formed by 50 synthetic instances and 164 real ones. In level 3, cluster 11 has the 50 synthetic instances and the 5573 real ones.

In all cases, hierarchical clustering impedes to improve these results in higher levels.

The conclusion is that modal clustering is not very sensible to detect new classes formed by very few instances, dispersed and overlapping other clusters. Clusters that contain a large portion of data tend to absorb surrounding disperse instances and smaller clusters. Detection requires higher density or to be far from surrounding clusters.

We have to notice that, for Autoclass, synthetic instances were generated following exactly the same model than Autoclass was searching for (multivariate gaussian) so the detection had to be necessarily simpler. For HMAC, the instances should have been created in other conditions.

5.5. Astrophysical interpretation

In this section, we tried a cluster interpretation using the results obtained in section 5.1. The identification of clusters with variable star types is done by comparing them with the Autoclass clusters identified by the expert.

In this case, HMAC (mtree) is not able to predict the class membership of the labeled dataset to identify clusters. But we know that these instances were extracted from OGLE Large Magellanic Cloud dataset. So we looked for these instances in it to analyze the HMAC clustering of the labeled examples.

We only show results with a significance above than 5%. We are only interested on level 3 (Table 5.6) that counts with 54 relevant clusters, but, as the results show a mixture of classes, we also show level 2 (Table 5.5) to verify if this mixture could have been avoided at a previous step. The results show that, in level 2, some clusters with a pure composition of just one type of stars can be found, but the large number of clusters obtained in this level, makes it more difficult to manage the analysis.

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
1							6.85			
3							10.27			
5				55.28	65.43					
6						10				
12							12.56	5.99		
14								5.41		
17							10.92			
20		14.08	14.28						100	97.96
21								5.33		
24								9.79		
25							8.8	8.62		
27	53.69		50							
33			7.14	35.54	18.51	47.5				
49	32.74	45.07	7.14							
124		33.8								
828			7.14							
3586			7.14							
3591			7.14							

Table 5.5. HMAC classification of labeled instances in hierarchical level 2 (percentage > 5%).

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
1							15			
2			7.14	94.68	87.65	60				
3			14.28		6.17	20	41.1	36.59		
4							15.17			
7						7.5	19.73	21.6		
11	57.12	49.29	64.28						100	98.24
12								5.26		
13						6.25		19.15		
22	33.73	45.07	14.28							

Table 5.6 HMAC classification of labeled instances in hierarchical level 3 (percentage > 5%).

As clusters are not labeled by its weight in the classification, Table 5.7 shows the most important clusters found.

Cluster	Instances	Cluster	Instances	Cluster	Instances
3	8119	14	421	57	63
11	5041	5	332	37	59
2	4956	10	329	79	55
7	4287	23	226	84	29
6	4277	26	183	50	28
9	2892	35	182	44	25
4	2012	33	169	38	25
1	1926	0	160	28	22
12	1669	34	137	32	18
13	1522	18	117	40	15
16	1121	17	104	53	14
20	574	21	100	70	14
8	545	30	97	75	12
22	530	36	96	114	11
19	424	31	88	47	11

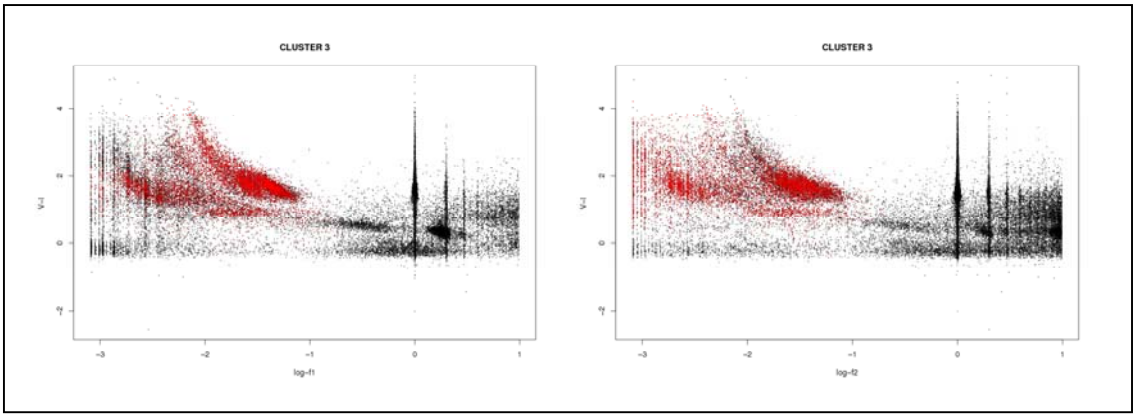
Table 5.7. Clusters ordered by the number of instances.

5.5.1 Preliminary analysis of clustering

A preliminary analysis of clusters 3 and 11 of hierarchical level 3 based on the visual observation of two dimensional plots are summarized in the following paragraphs.

Cluster 3 - OSARGS + Mira and semirregular stars + ellipsoidal stars

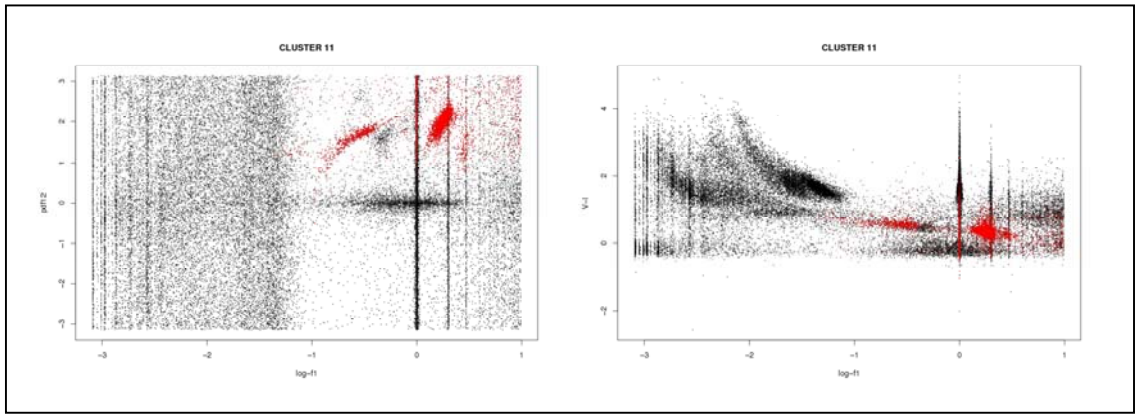
Cluster 3 is the cluster of the highest weight in the classification. This first result is deceiving because cluster 3 gathers instances belonging to different variable stars types that are visually well separated in different groupings. We include two 2-D plots representing the V-I colour as a function of the logarithm of the first (Figure 5.1) and second (Figure 5.2) frequencies.



Figures 5.1 and 5.2. Plots of the location of instances in cluster 3-level3 in the log-f1 - (V-I) color index plane and in the log-f2 - (V-I).

Cluster 11 - RRLyrae + Cepheid stars

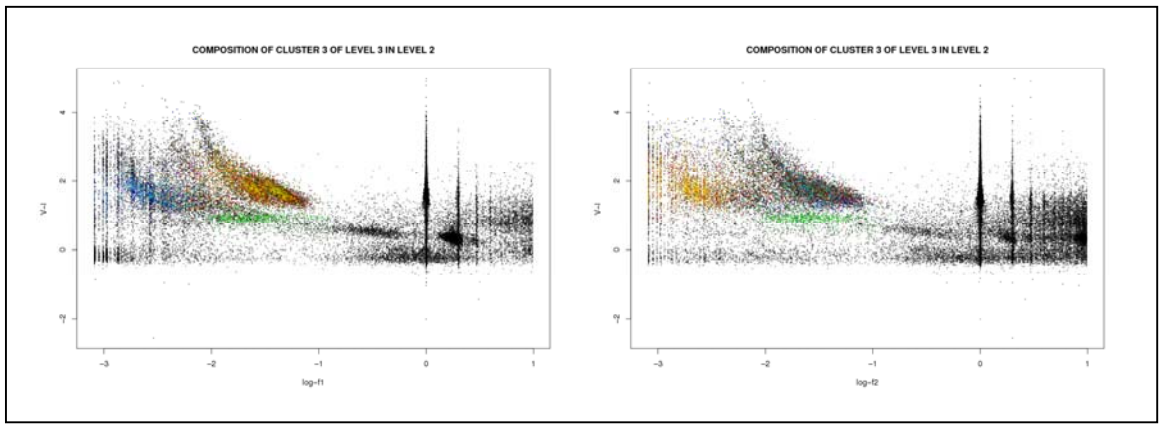
Cluster 11, the second with greatest weight, presents the same problem than cluster 3. It is grouping RRLyrae stars and cepheids stars, types of variable stars visually well separated (see Figures 5.3 and 5.4).



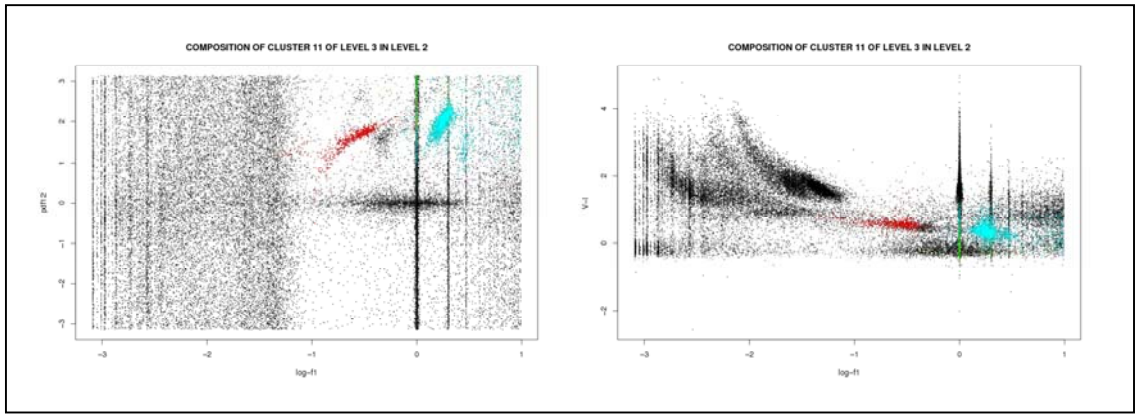
Figures 5.3 and 5.4. Plots of the location of instances in cluster 11 in the log-f1 - pdf12 and log-f1- (V-I) index plane

As the results were not good enough we studied the evolution of groping in previous hierarchical levels that conducts to this situation.

For this analysis, we focused in cluster 3 of level 3 and we saw that this cluster came from 13 main clusters in level 2 (clusters with more than 20 instances that are the 92.4% of the total). We removed small clusters to simplify the analysis (in total, cluster 3 of level 3 is formed by 544 clusters in level 2). The same happened with cluster 11 of level 3. This cluster is formed by grouping 172 clusters in level 2, but only 8 clusters of them have more than 20 instances (96,2%). Figures 5.5 and 5.6 show the subclusters of cluster 3 , and Figures 5.7 and 5.8 the subclusters of cluster 11 in different colours. Although some instances overlap, it is clear that HMAC has grouped in level 3 clusters of level 2 that should have been left alone.



Figures 5.5 and 5.6. Plots of the location of instances of clusters in level 2 of cluster 3 in level 3 in the log-f1 - (V-I) colour index plane and in the log-f2 - (V-I) in different colours.



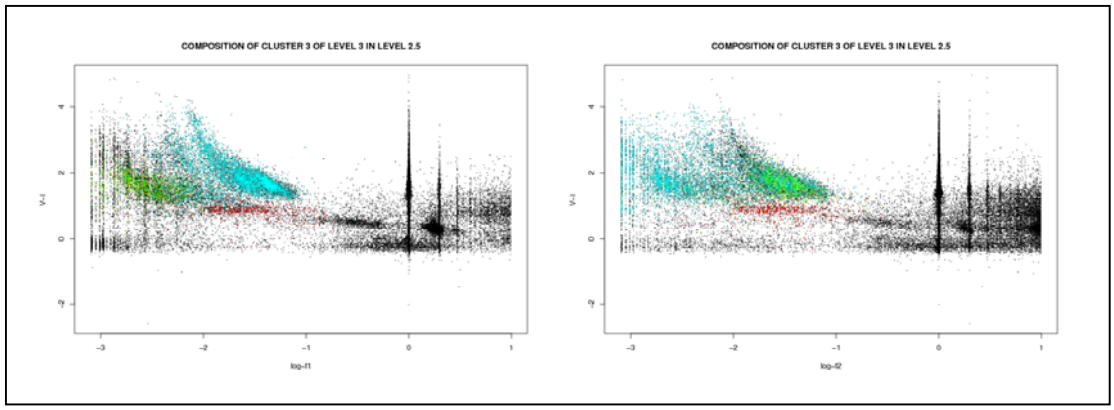
Figures 5.7 and 5.8. Plots of the location of instances in cluster 11 in the log-f1 - pdf12 and log-f1- (V-I) index plane

At this point the decision to deal with is if it reasonable to work in level 2 with so many quantity of clusters (4306 clusters in total, 174 with more than 5 instances) or if it is better to try a next level of hierarchical clustering in other conditions.

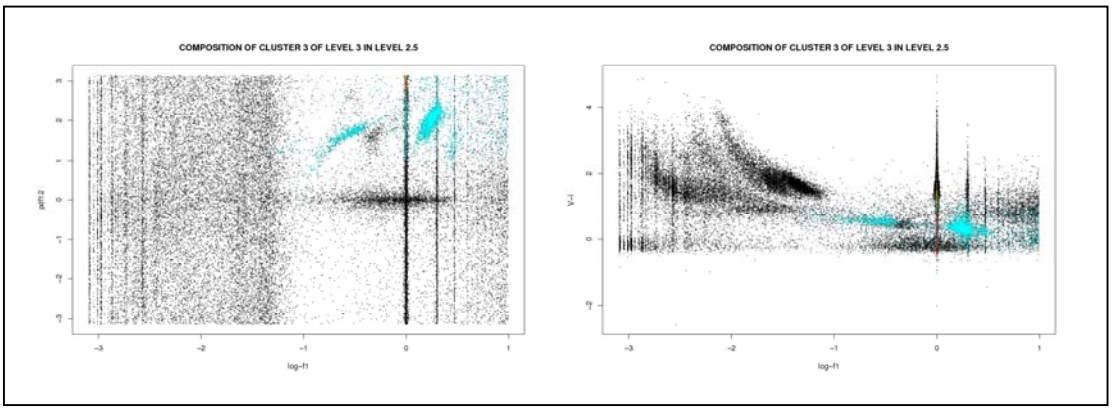
There are two possibilities to reduce the number of clusters:

- to try a sigma step smaller than the used (0.3388 in level 2, 0.5083 in level 3)
- to specify a merge parameter in HMAC algorithm.

We tried both options. In option 1 we executed HMAC(mtree) to perform a hierarchical clustering but we specified a bandwidth sequence (0.17, 0.34, 0.42, 0.51) including the intermediate value 0.42. The results were again deceiving, because we split out some classes in cluster 3 (Figures 5.9 and 5.10) but not in cluster 11 (Figures 5.11and 5.12).



Figures 5.9 and 5.10. Plots of the location of instances of clusters in level with bandwidth value 0.42 of cluster 3 in level 3 in the log-f1 - (V-I) colour index plane and in the log-f2 - (V-I) in different colours



Figures 5.11 and 5.12. Plots of the location of instances of clusters in level with bandwidth value 0.42 of cluster 11 in level 3 in the log-f1 - pdf12 and in the log-f1 - (V-I) in different colours

It seems that, modifying only sigma values of the kernel, the results indicate the need of selection of clusters in different hierarchical levels according to the clustering obtained. It supposes an injection of knowledge about what one wants to obtain. To select just a hierarchical level by the number of prominent clusters it contains has shown to be very inefficient.

In option 2, we tried to execute HMAC (mtreesep) in the same conditions as the used previously (section 5.1) but specifying parameters for merging *(-v 0.5, -c 0.99 -w 2*). This is a two step merging in level 2, one based on separability between pairs of clusters, the other on coverage rate. With these parameters, clusters with separability smaller than 0.5 will be merged and clusters with relative size lower than 1% of data are considered outliers and merged to a cluster. What we wanted to avoid is that prominent clusters be clumped, fact that happened increasing bandwidth value in level 3. With the coverage rate, we prevented that outliers, that generally have a high separability, forced to increase too much the bandwidth value to join them to a prominent cluster.

However, mtreesep was unable to execute with these parameters. The algorithm code has enormous problems with memory allocation because it stores in memory all the data of all hierarchical levels without any optimization. Although we tried to correct major problems, we gave up to re-code the algorithm.

So, in order to reduce computation time and memory requirements, we did not process the original dataset but the results of previous executions. We used the modes obtained in level 2 as dataset input and we created a weight file with number of points of each cluster. The density estimate now uses this weight value for each term in the summatory. This is not an approximation to accelerate execution, this is an exact execution of the algorithm since we were using a previous clustering produced by the same method.

HMAC reduced initial clustering with 4306 clusters to 2838 with a sigma value 3.749834e-001. After that, it merged clusters with the conditions specified. Then the algorithm reduced from 2838 clusters to 25, 24 clusters were left as they

were formed and cluster 0 absorbed the rest of small clusters creating the biggest cluster (8852 instances). Table of contingence 5.8 shows that this merging produced very bad results, results even worse than the obtained without merging and sigma value 0.5083199.

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
0	6.7	33.8	28.57	95.33	87.65	61.25		26.43		
1	55.82	15.49	64.28						100	98.16
2							24.95	15.5		
5							16.31	6.39		
6						7.5				
8								10.09		
9							12.56			
10							7.5	5.41		
11	33.51	45.07	7.14							
14							8.31			
21						16.25	5.38			
24								15.39		

Table 5.8. Class membership for labeled instances after merging: bandwidth=3.749834e-001 separability=0.5, coverage rate=99%(percentage >5%)

Other merging parameters values were tested and the results were:

1. separability 0.25 and coverage rate 1: In this clustering, only separability is being considered. The algorithm reduced the number of clusters from 2838 to 1623 clusters
2. separability 0.25 and coverage rate 0.995. The final number of clusters was 38.
3. separability 0.25 and coverage rate 0.95. The final number of clusters after merging was 6.
4. separability 0.75 and coverage rate 0.99. The final number of clusters is 20.

The first and second case show that there were many clusters considered as outliers at this level and merging by coverage rate was not efficient. On the other hand, merging by separability was not enough to reduce the number of clusters. Merging seems not to be a solution at very early stages of hierarchical modal clustering where there are many clusters still growing.

5.5.2 Final analysis of clustering

Finally, we tried a cluster interpretation from the results obtained from a new execution of the algorithm. HMAC (mtree) was configured to perform a hierarchical clustering with the following parameters: bandwidth sequence (1.694400e-001, 3.388799e-001, 0.37, 0.40, 0.42, 0.45, 5.083199e-001, 6.777598e-001, 8.471998e-001, 1.016640e+000, 1.186080e+000, 1.524960e+000). The initial values for sigma used previously were maintained

but we introduced some more intermediate values between the sigma values that generated level 2 and level 3 in previous executions.

In this conditions, the dendrogam obtained is shown in table 5.9.

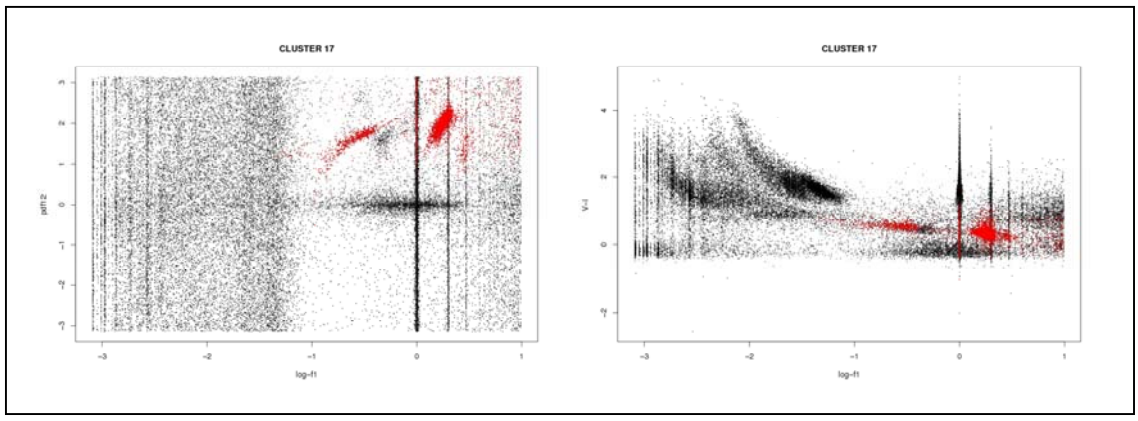
Level	1	2	3	4	5	6	**7**	8	9	10	11
Band-width	0.1694	0.3389	0.3700	0.4000	0.4200	0.4500	**0.5083**	0.6778	0.8472	1.0167	1.1860
# clusters	36289	4306	3958	2281	1782	1215	**494**	59	14	4	2

Table 5.9. Dendrogram obtained by HMAC search over the OGLE LMC dataset

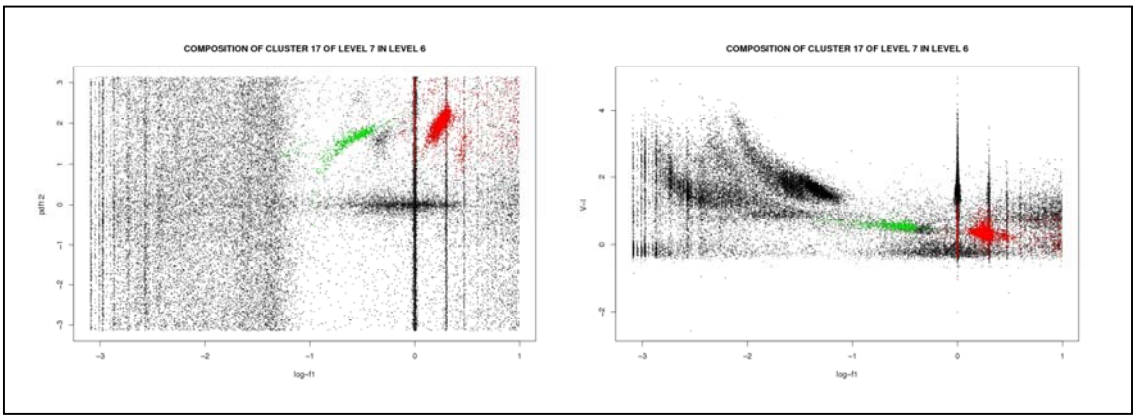
We tried a clusters interpretation taking level 7. The sigma value applied in this level is the same than the selected in section 5.1 but the number of clusters obtained now is different because of the bandwidth sequence used. The results in these new conditions are better than before.

Cluster 17 RRLyrae and Cepheid stars

Cluster 17 (Figures 5.13 and 5.14) still combines RRLyrae and fundamental mode Cepheids. This cluster is formed by 12 clusters from previous level but only two of them form the 97% of data. Figures 5.15 and 5.16 show how these types of stars are separated in level 6 (sigma 0.45). It is necessary to descend to level 1 to separate spurious values log-f1 = 0 that are joined to RR Lyrae in another differentiated cluster.



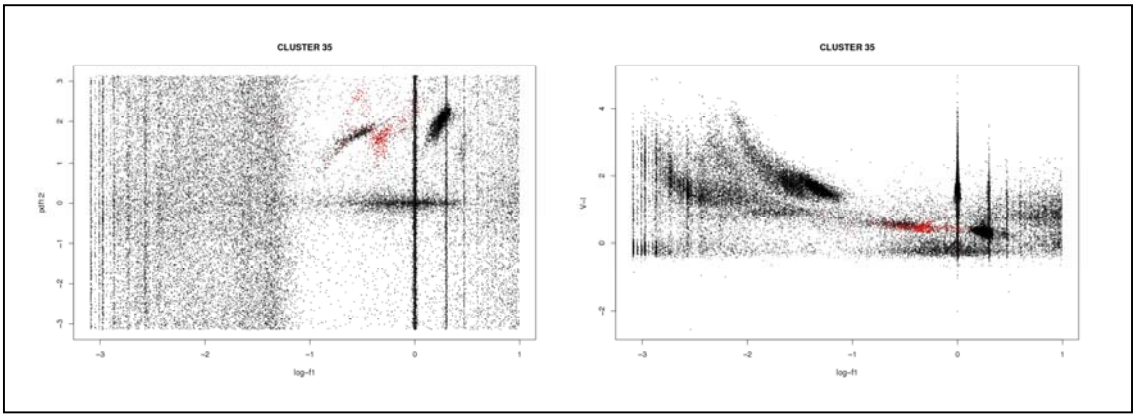
Figures 5.13 and 5.14. Plots of the location of instances of clusters 17 in level 7 in the log-f1 - pdf12 and in the log-f1 - (V-I)



Figures 5.15 and 5.16. Plots of the location of instances of clusters 17 in level 6 in the log-f1 - pdf12 and in the log-f1 - (V-I) where rrlyrae and cepheid stars are separated.

Cluster 35: cepheids

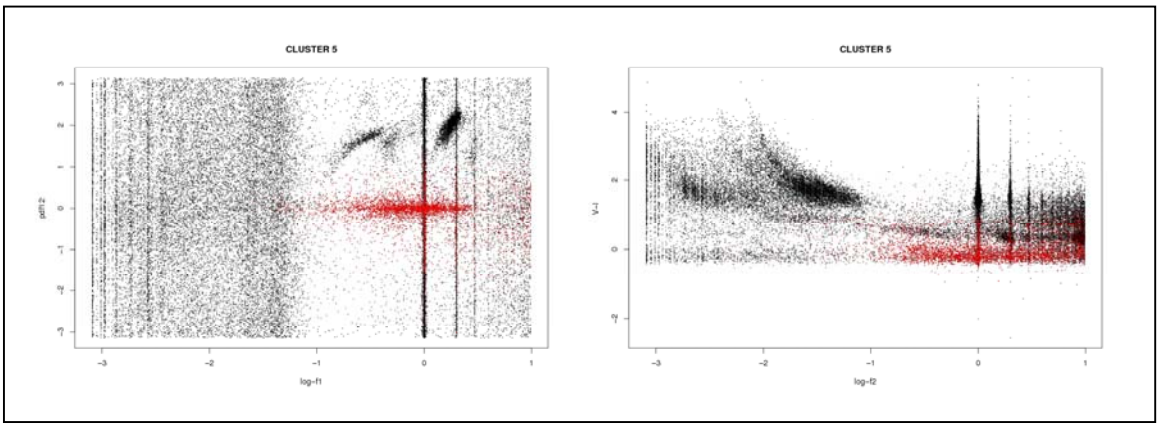
Cluster 35 (Figures 5.17 and 5.18) contains first overtone pulsators cepheids.



Figures 5.17 and 5.18. Plots of the location of instances of clusters 35 in level 7 in the log-f1 - pdf12 and in the log-f1 - (V-I)

Cluster 5: eclipsing binary systems

This is the greatest cluster and represents eclipsing binary systems. The cluster also gathers instances with spurious data in the log-f1 and log-f2 attributes. See figures 5.19 and 5.20.



Figures 5.19 and 5.20. Plots of the location of instances of cluster 5 in level 7 in the log-f1 - pdf12 and in the log-f1 - (V-I)

Cluster 6: ellipsoidal stars

Figure 5.21 shows a plot of the cluster 6 identified as containing ellipsoidal stars.

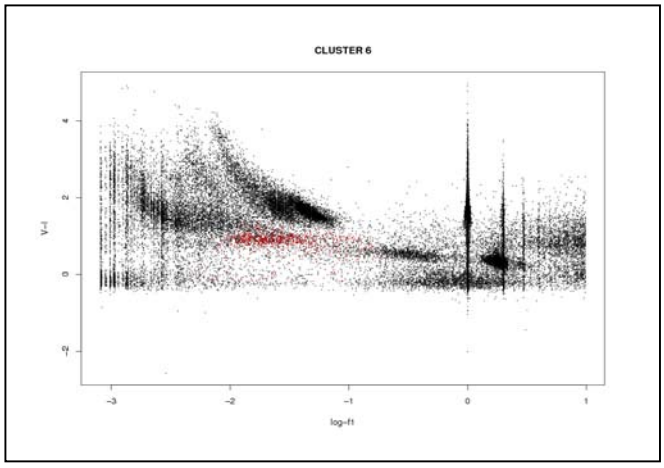


Figure 5.21. Plots of the location of instances of cluster 6 in level 7 in the log-f1 - (V-I)

Cluster 32: BE stars

Cluster 32 (Figure 5.22) occupies the locus of BE stars according to previous identification in Autoclass clustering.

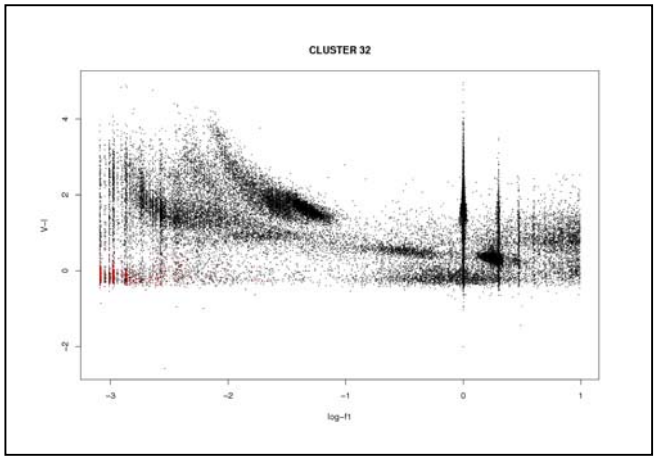


Figure 5.22 Plots of the location of instances of cluster 32 in level 7 in the log-f1 - (V-I)

Clusters 3, 21, 12, 1, 27, 16, 7: OSARGS (and mira and semirregular stars).

The identification of these clusters is complicated. Sometimes mira and semirregular stars appear together and it is difficult to decide which one dominates. Clusters are mentioned and grouped looking for similarities with the classification proposed for these type of stars in Autoclass clustering.

Cluster 3 (Figures 5.26-5.27) has a mixture of classes although it seems that OSARGS predominate over mira and semirregular stars.

Cluster 21 (Figures 5.28-5.29), 27, 16 and 7 have OSARGS and correspond with cluster 6 of Autoclass.

Cluster 12 (Figures 5.30 and 5.31) and 1 have OSARGS and correspond with cluster 0 of Autoclass

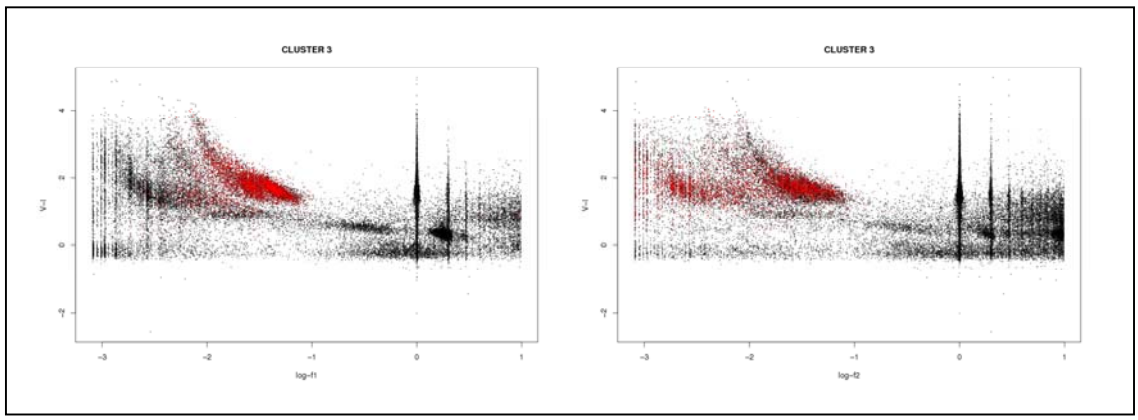
Cluster 10, 14 and 19 occupy the location of OSARGS although log-f2 attribute in both clusters present spurious data.

Cluster 20 and 26 occupy the location of OSARGS although log-f1 attribute presents spurious data.

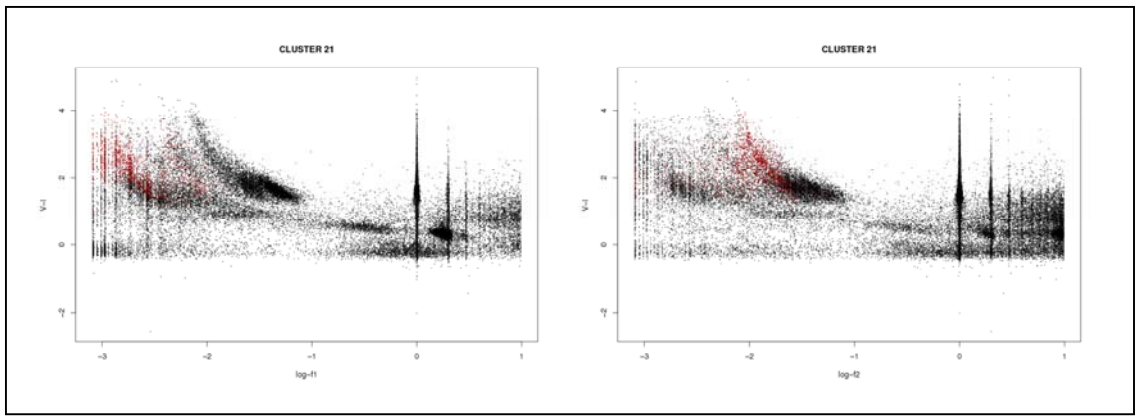
As there are several clusters representing the same type of stars, here we show how they merge in a higher hierarchical level (Table 5.10). The agglomerative process merges these clusters but also with other clusters not related with OSARGS.

Level 8	1	1	3	3	3	9	13
Level 7	1	27	3	7	16	12	21

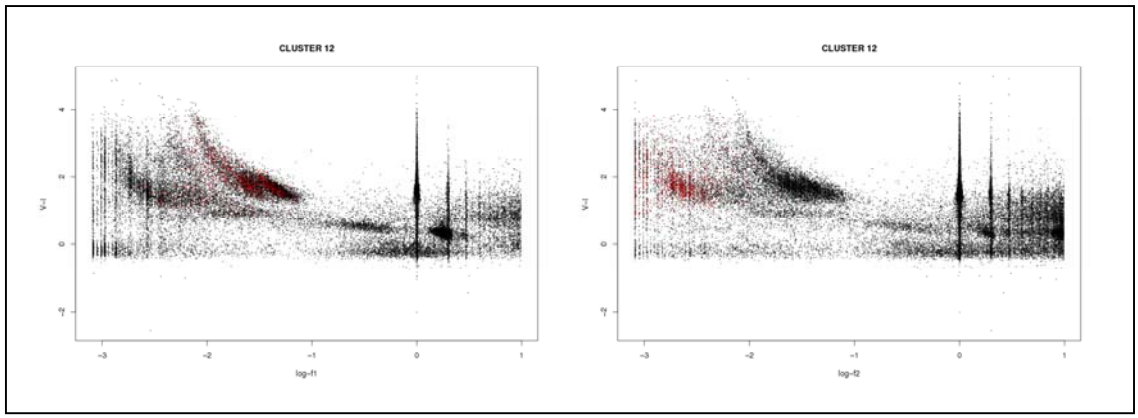
Table 5.10. Cluster merging of OSARGS in next hierarchical level



Figures 5.26 and 5.27. Plots of the location of instances in cluster 3 in the log-f1- (V-I) and log-f2 - (V-I)



Figures 5.28 and 5.29. Plots of the location of instances in cluster 21 in the log-f1- (V-I) and log-f2 - (V-I)



Figures 5.30 and 5.31. Plots of the location of instances in cluster 12 in the log-f1- (V-I) and log-f2 - (V-I)

We notice that, in section 5.1, OSARGS and mira and semirregular stars were grouped in just one cluster, and now, with exactly the same bandwidth value, these stars appear divided in several clusters. In relation with RRLyrae and cepheid stars happens the same. We tried sigma value 0.42 and these types of stars were grouped, now with the same value, they are separated. This makes think that initial strategy of bandwidth sequence was erroneous. The step between one sigma value and the next one has to be smaller enough as it is now. But we have not completely avoided the need of selecting clusters of different hierarchical levels.

5.6 Experiments with a reduce attribute set.

Our next experiment was to apply HMAC(mtree) to find a solution using only 5 of 13 attributes (log-f1, log-af1h1-t, log-crf10, pdf12, V-I). The search was configured to perform a hierarchical clustering over the total dataset and using these 5 attributes. The rest of parameters were configured with its default values (step size of the bandwidth sequence=0.1, maximum bandwidth in sequence = 2.0). This invocation of HMAC search took 1 day 51 minutes 53 seconds and gave a dendrogram of 5 levels. Tables 5.11, 5.12, 5.13 show the results obtained in level 1 (bandwidth 1.694400e-001, clusters 1471), 2 (bandwidth 3.388799e-001, clusters 46), and 3 (bandwidth 5.083199e-001, clusters 9). Again, we only show results with a percentage of instances belonging to a cluster greater than 5%. With 5 attributes the clusters have a similar composition to the obtained with 13 attributes. Although level 1 has produced 1471 clusters, most of the labeled instances show already the preferences of grouping that will maintain in higher levels, except lpv stars that are dispersed in many clusters.

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
4							5.22			
5	5.78	69.01	21.42						98	99.76
6							6.03	6.72		
7							7.66			
9							8.48	7.42		
10				91.81	82.09	47.5				
23								9.57		
25	58.64	14.08	50							
35							8.64			
40					8.02	36.25	6.03			
80	26.19	11.26	14.28							
111							6.03			
120							6.52			
187							8.64			
491			14.28							

Table 5.11. Contingency table using clustering of level 1 of the dendrogram (percentage> 5%)

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
0					8.64	42.5	43.55	44.82		
1			14.28				20.22	11.7		
2								12.13		
3						6.25	14.51	24.31		
4	95.5	94.36	85.71						98	99.8
5							7.66			
7				92.82	82.09	47.5				
8		5.63								
13							8.64			

Table 5.12. Contingency table using clustering of level 2 of the dendrogram (percentage > 5%)

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
0					11.72	48.75	67.21	70.93		
1			14.28				29.03	13.38		
2		5.63						15.53		
3	95.5	94.36	85.71						98	99.8
5				92.94	82.71	47.5				

Table 5.13. Contingency table using clustering of level 3 of the dendrogram (percentage > 5%)

The conclusion is that these five attributes are relevant for the clustering and could be used instead of the original thirteen.

5.7 HMAC applied to cluster the database of labeled examples.

Finally, we used HMAC to find a solution over the sum of all test datasets (10063 instances). The processing conditions were the same as in the previous experiment but using again the original set of 13 attributes. The algorithm found 8 hierarchical levels. We focused our attention in level 4 and the analysis of the results (Table 5.14) implies that the classification is very similar to the one obtained on the OGLE LMC dataset. The main major clustering structures are retained in both datasets. Both classifications in the level of comparison are not using the same sigma value (0.5429795 in level 4 over the labeled examples against 0.5083199 in level 3 over the OGLE dataset).

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
0						10	18.27	59.34		
1								14.84		
2								6.61		
3								6.69		
9			21.42	98.25	95.06	80				
12							7.5			
17	58.49	40.84	64.28						100	98.24
26							18.43			
29							20.88			
39	32.9	53.52	7.14							
51			7.14							
53		5.63								
56							14.35			
64							9.78			

Table 5.14: Clustering structure of level 4 using only labeled datasets. Only percentages above 5% are shown for clarity

5.8 Effect of considering log-normal attributes

We repeated the experiment using a linear scale for attributes 1, 2, 3, 4, 5, 6, 7, 8, 9. HMAC (mtree) was configured to perform a hierarchical clustering, step size of the bandwidth sequence=0.2 and maximum bandwidth in sequence = 3.0.

The results show that the log-transformation is very useful for HMAC clustering. With the linear scale, in level 1, HMAC found 131 clusters, but only to analyze the percentage of instances of each cluster was enough to verify that the clustering was not very efficient. As can be seen in Table 5.15, cluster 0 (sigma value= 5.347489e-001), with greater weight (56.8% of instances), gathers almost all type of stars.

Cluster	cep	dmcep	ptcep	ecl	new-ecl	ell-ecl	ell-ell	lpv	rrd	rrlyr
0	6.77	5.63	21.42	9.24	15.43	75	93.8	99.59		
4				87.15	80.86	25				
5										47.41
8		39.43	7.14						98	41.2
10	88.72	54.92	71.42							
24										8.01

Table 5.15 Clustering structure using the labeled dataset

This is due to that, without the log-transformation, the data points density between 0-2 range values increase considerably, and clusters that are mainly in

that range values have a lower separability. The bandwidth step should be very reduced to analyze this range values.

5.9 Comparison with Hipparcos dataset

We applied HMAC to the Hipparcos dataset (2498 instances) with exactly the same attribute information as the original OGLE dataset. HMAC (mtree) was configured to perform a hierarchical clustering, step size of the bandwidth sequence=0.1 and maximum bandwidth in sequence = 2.0.

There is no way to verify the clustering goodness because this implementation of HMAC can not be used to predict the class membership of other labeled instances to validate the results. So, we only show plots of the location of clusters and instances in different colors for the most relevant attributes. (See Figure 5.32). Clustering interpretation can be only carried out by a domain expert or comparing to Autoclass results.

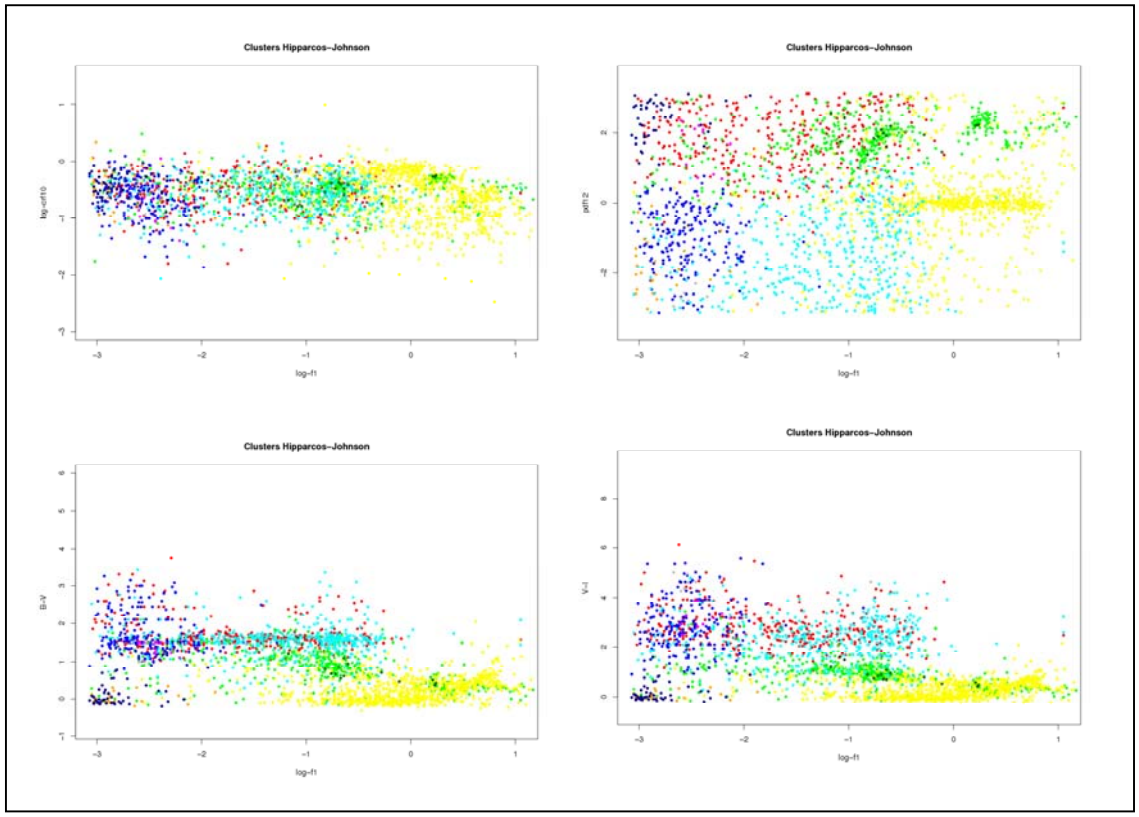


Figure 5.32. Main clusters obtained over Hipparcos dataset in the most relevant attributes(log-f1 vs log-crf10 pdf12, B-V, V-I) in different colors.

6. A clustering combination method: Autoclass and HMAC

In this chapter, a combination of the two clustering algorithms, taking the best of both approaches, is undertaken to try to improve clustering performance. As clustering is an optimization procedure based on a specific clustering criterion in which it bases, a combination of two procedures can be regarded as a technique that optimizes according to multiple criteria.

On previous chapters, HMAC has shown to be expensive computationally to process large datasets, at least, without any code optimization. Autoclass has shown to work well but, perhaps it is a little restrictive due to the need of fitting a model. However HMAC can solve this problem. The combination of both procedures can be use to override the restrictive behaviour of Autoclass and to reduce data points to HMAC clustering and, consequently, reduce processing time.

The idea is to combine clustering in two sequential steps. First to apply Autoclass to find a first basic clustering, and then, to apply HMAC to take advantage of its agglomerative properties to merge clusters without having into account a restrictive model fit for modifying the initial result.

Now Autoclass has to be configured to produce more clusters than the ones produced in section 4. A good number of cluster for HMAC input could be 200. This can be obtained if Autoclass does not use a covariant model for data, this is, if considering independent attributes. When it is clear that the attributes are correlated, Autoclass will need more normal distributions to fit data. The input dataset to HMAC is formed by the resulting mean values for each attribute of each Autoclass cluster. And, as each cluster is formed by different number of instances it is also necessary to introduce to HMAC the weight for each cluster.

Then, Autoclass was configured with the following parameters 18h processing (*max_duration = 64800*), without random initializations, (*force_new_search_p = true, start_fn_type = "block", randomize_random_p = false*) starting with list (*start_j_list = 150,200,250,300*), and all atributes following a single normal distribution (*single_normal_cn 0 1 2 3 4 5 6 7 8 9 10 11 12*) and applied over the total dataset (13 attributes, 43351 cases).

In these conditions, Autoclass found 259 clusters on try 31 of 119. As mentioned previously, these clusters were the dataset input to HMAC.

After that, HMAC (*mtree*) was configured to perform a hierarchical clustering with the following parameters: bandwidth sequence (1.112001e-001,2.224002e-001, 2.780002e-001, 3.336002e-001, 3.892002e-001, 4.448003e-001, 5.004003e-001, 5.560004e-001, 6.672005e-001, 7.784005e-001, 8.896006e-001, 1.000801e+000, 1.223201e+000).

Now, HMAC search only took 8 seconds to process the dataset and the dendrogam obtained is shown in table 6.1.

Level	1	2	3	4	**5**	6	7	8	9	10	11
Band-width	0.11120	0.22240	0.27800	0.33360	**0.38920**	0.44480	0.50040	0.55600	0.66720	0.77840	0.88960
# clusters	259	256	235	192	**127**	82	53	36	19	5	3

Table 6.1. Dendrogram obtained by HMAC after Autoclass clustering

Again, outliers are a problem to identify the hierarchical level of interest. After a detailed inspection, the most interesting level is 5.

Now, not all the resulting clusters are shown but just a few cases to show how the combination of methods has worked:

- RR Lyrae: Autoclass found 3 clusters that contained RRLyrae stars, now, only one is found. This cluster does not contain spurious data (log-f1=0.3) as HMAC cluster does.

- Ellipsoidal stars: Visually this cluster presents more defined borders and better compactness than the one obtained by Autoclass.

- Cepheids: There are also two clusters separating first overtone pulsators and fundamental mode Cepheids.

- Other clusters representing OSARGS and Mira and semiregular are more complicated to interpret.

As a conclusion, this combination procedure seems to give good results:

- To merge clusters for Autoclass. If data do not behave exactly as the specified model in Autoclass search, there will be several distributions to fit data that have to be merged.

- To reduce large datasets for HMAC search. Data reduction have a high impact in processing time without great differences in final results.

7. Performance evaluation

Up to now, we have shown the results of application of two different algorithms for unsupervised clustering. The identification of major clusters obtained has been performed applying the domain expert knowledge and using the labeled datasets. But not all the clusters have been identified and some types of stars are characterized by several clusters. In addition, there is a remain problem of global clustering evaluation, which one of the two clustering algorithms gives best results. This is not an easy question to answer when dealing with unsupervised clustering.

7.1 Comparative evaluation of different clustering algorithms

Unsupervised clustering counts on its difficulties the evaluation of the results obtained according to a quality criteria global shared among all algorithms that allows to compare them in the same conditions.

One evident global criteria to verify the goodness of the clustering could be by its ability to classify correctly labeled instances. This ability could be measured by the error of classification. But in our study we have seen that this is not an easy task. If we had enough and good labeled instances to use in this task, why to use unsupervised methods instead of supervised ones for clustering? In our case we did not have many labeled instances and, for some classes, they were totally insufficient (PTcep 14 instances, RRLyrae 50). In some cases, a cluster gathers different classes so, with which criteria should we evaluate this cluster? Moreover, sometimes the algorithm is not able to predict a classification of new cases. For that task, the algorithm requires to model each cluster with a parametric function. In this study we have seen that Autoclass has that ability but HMAC does not so, as an example, we have not able to extract any conclusion from the results of application of HMAC over Hipparcos dataset.

But, in addition, to evaluate a clustering intervene other subjective criteria. In the context of GAIA, it seems that one of the objectives is to use unsupervised methods for new classes discovery. It supposes to detect anomalous instances but sharing some characteristics to group them. And, in general, these clusters are formed by very few instances compared to the expected clusters. So, clustering algorithms that give few importance to small clusters and these clusters can be absorbed by neighbouring clusters with large quantity of data, will be evaluated worse than other that considers equal big and small clusters. With this criterion, Autoclass has more probabilities to detect new and small classes than HMAC. HMAC has also possibilities to detect them, but as it is an agglomerative algorithm, in the level where a new small class could be detected, there will be also a lot of small clusters growing, so the identification could be more difficult.

In the case in which we do not want to discover new classes but question actual classification, the use of labeled datasets would never be a way to verify clustering because traditional classification is what is being questioned. One can use the new classification as a departure point to construct new models or theories. And how to evaluate this with objective quality parameters without using the most important factor that is the domain knowledge of the expert.

Another possibility is to use the different internal quality criteria that each algorithm utilizes. Autoclass tries to maximize the marginal likelihood of data over the parameters of a distribution, and the quality of the results are measured by the logarithm of this parameter. HMAC uses modal expectation maximization to find the local maxima of a given distribution. But these measures are only valid to compare different executions (and only in the first case) of the same method, not to compare the results among them. The same happens with other internal quality criteria that other algorithms use as the minimization of the intraclusters distances or maximization of intercluster distances. The differences between these internal quality measures make them practically impossible to objectively compare clustering algorithms.

We could also generate some synthetic datasets according some probability distributions, and ask to the different algorithms to find the best solution. In this case one could have available some statisticals of error rate to compare results. But, can we be sure that the data of our domain follow that distribution? This method is only valid on a theoretical basis but not in a real context. Moreover, the criteria that could be valid for a domain could not be valid for other.

Despite these considerations there exist some validation indices in the literature that try to evaluate clustering results using objective criteria. These are classified into three groups:

1. **Internal validation indices**: These indices determine if the structure is intrinsically appropriated for the data and is based on calculating properties of the resulting clusters, such us compactness, separation, roundness. These methods does not require additional information about the data.

   Some of these indices are **Dunn's indices** defined as the ratio between the minimum distance between two clusters and the size of the largest cluster; the **Silhouette index** defined as the average, over all clusters, of silhouette width of their points; the **Hubert's correlation with distance matrix** that measures the similarity between the points to be grouped; **Davies-Bouldin index** that is a function of the ratio of the sum of within cluster scatter to between cluster separation.

2. **External validation indices**: Compares the clustering to an a priori structure and tries to quantify the match between the two. External validation corresponds to a kind of error measurement either directly or indirectly. For example, the **Rand index** can be used to match between two clustering

measuring the proportion of pairs of vectors that agree by belonging either to the same cluster or to different clusters. Other indices are **Jaccard coefficient**, that measures the proportion of pairs that belong to the same cluster in both partitions, relative to all pairs that belong to the same cluster in at least one of the two partitions, and **Folkes and Mallows** index measures the geometric mean of the proportion of pairs that belong to the same cluster in both partitions, relative to the pairs that belong to the same cluster for each partition.

3. **Relative validation** is based on comparison of partitions generated by the same algorithm with different parameters of different subsets of data. These methods do not require either additional information. Some indices are **Figure of merit** is designed to aid decision of which clustering method is appropriate and how many clusters are optimal or **Stability** that measures the ability of a clustered data set to predict the clustering of another data set sampled from the same source. But these indices perform well for synthetic data however there is not guarantee that this index and other will be optimal for real data, and the characteristics of data can affect performance in unknown ways.

Experimental tests over these indices indicate that the performance of validity indices is highly variable. For complex models or when a clustering algorithm yields complex clusters, both the internal and relative indices fail to predict the error of the algorithm. Some external indices appear to perform well, whereas others do not. The conclusion is that one should not put much faith in a validity score unless there is evidence, either in terms of sufficient data for model estimation or prior model knowledge, that a validity measure is well-correlated to the error rate of the clustering algorithm.

From these indices, **stability** could be an interesting index to try because it expresses that the results of a good clustering algorithm are stable with respect to the sampling process, this is that they do not change much if one draw another sample or add or delete some point from the dataset. This is in relation to the GAIA mission that will gather data along the mission lasts and the dataset will grow accordingly. Results from consecutive analysis must be stable in relation to the number of clusters and meaning. Stability must also be maintained in the presence of noise, so this is an important property of clustering since astrophysical data used seem to be noisy. Consequently, stability is an indication if whether the model proposed by the algorithms fit to the data or not in these conditions and not only in a simple test.

Another index that could have been used for analyzing Autoclass results in section 4.1 is the **Rand index** or the **Adjust Rand** index, a variant version of the former. The statistic is based on the relation of every pair of cases in the study and whether these relations differ between two solutions. This avoids the need to specify one solution as correct, and then assess how well the second solution reproduces the first. The index takes a value of 1 for perfect agreement between two clustering solutions, and a value of 0 if agreement is equal to that

expected solely due to chance. There is a wider range of values that the Adjusted Rand index can take compared to the Rand index increasing its sensitivity.

Just as an example, these two indices have been applied to the results obtained in section 4.1 where the two more different Autoclass solutions were compared to see the impact of randomess in results. The match between clusters was done by hand and small clusters were removed to calculate these indices although most of the instances were used (85.7%, 37171 instances). The values obtained were:

$$\text{Rand Index}_{\text{between solution 3 and 5}} = 0.9859$$
$$\text{Adjusted Rand Index}_{\text{between solution 3 and 5}} = 0.8620$$

Taking into account that both indices have 1 for prefect agreement, the values obtained confirm the qualitative analysis performed in section 4.1 and both solutions are very similar. However, qualitative analysis perhaps supply better knowledge about where are the discrepancies and if they can be afforded or not, than a simple figure.

None of these indices will be applied more deeply to compare our clustering results. Only a qualitative analysis of each one of the clustering methods applied is shown in the following sections with the information that each algoritm supplies.

7.2 Autoclass evaluation

Autoclass is a model based clustering. In this approach it is assumed that the data are following a mixture of underlying distributions in which each distribution represents a different cluster. In our case we have supposed that each cluster is characterized by a multivariate normal distribution. This means that the clusters are ellipsoidal, centered at the means $\mu_{k,}$ and the covariances matrix Σ_k determine other geometrical characteristics as the axis orientation.

If data do not follow this model, if densities of individual clusters are multimodal and cannot be accurately modelled by basic parametric distributions or if data are contaminated by spurious detections, as happens with some attributes in OGLE LMC dataset, Autoclass will produce more cluster than expected due to the need of model each real cluster by a set of multivariate normal distributions.

Autoclass supplies several parameters that permit to analyze clustering and to verify how good is the clustering obtained. The first indicator to verify if the classes are well separated is the **vector of probabilities of each instance** to be member of a cluster. If a great number of instances of a cluster have a high percentage of being member of other cluster, then one could conclude that this

classes are not well separated. Another parameter is the **class strength** defined as the geometric mean probability that any instance belongs to a class. It thus provides a heuristic measures of how strongly each class predicts its instances.

There is also information on the importance of the individual attributes, both for the classification overall and for each class. The divergence measure is the **Kullback-Leibler distance (or relative entropy**). This is a useful measure of distance between data distributions because it takes into account both the centre of the distribution and the variability of the data around the centre. However, it is not a true metric because distances are not symmetric. Thus the distance from distribution Q to distribution P does not necessarily equal the distance from P to Q. The D_{KL} of P to Q for continuous variables is defined as

$$D_{KL}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx$$

where p and q denote the densities of P and Q. D_{KL} is 0 when the two distributions are identical.

The modelled distribution of each attribute for each class is also supplied, ranked by the **attribute relative importance** in describing the class. Lastly, the **overall divergence** of each class from the overall distribution of cases, calculated as the summed D_{KL} over all attributes, is reported. However, for covariant models as the used here, a single attribute parameters are mixed with those of all of the other attributes through the covariant terms, so the calculus is not easy. For this reason, the relative importance of each attribute is just the term cross entropy divided by the number of attributes modeled in the term because all attributes in a multiple term have common influence, so consequently, all of them have the same value. The real interaction among them can be seen in the covariance matrix.

Table 7.1 shows all these indices: relative class strength of the class, the class divergence, and the rest of the values indicate the number of standard deviation separating the class-attribute mean from the global mean. Positive values are the class-attribute distribution on average greater than the global distribution, while negative values indicate the opposite. Grey rows are the distributions with minor class divergence or Kullback-Leiber distance between the class and the global distribution. The attribute influence of these classes have a $D_{KL} < 1$ for each attribute. (This value has been chosen arbitrarily). Obviously, greater clusters will have minor D_{KL} values.

According to the class divergence as it is calculated by Autoclass for covariant models, clusters 0 (OSARGS), 1, 2, 3, 4, 6 (OSARGS), 7 (Ellipsoidal), 8 (Mira and Semirregular), 9, 11, 12, 15 have distributions not very different from the overall distribution although they can be different from other classes.

Cluster 5 is the class with the highest class strength that means that is the cluster that better predicts its instances. The smallest cluster, cluster 47, is the

cluster most divergent from the global distribution. So, divergence of a class does not necessarily imply class strength, as it is shown in these two examples.

Interpretation of rest of figures in class-attribute combination can be done individually in the following way. For example, class 13 (BE stars) represents variable stars with low log-f1, B-V, V-I mean values compared to the global classification; Cluster 26 shows stars with very high log-f1 values that diverge from the global distribution whereas the rest of mean values are near of the overall distribution ones. Clusters with spurious values in the log-f1 and log-f2 attributes can also be easier detected, for example, cluster 4 with log-f2 value equal to 0, or Cluster 33 with log-f1 values equal to 0.

Class	Rel class strength	Class divergence	Attr 0	Attr 1	Attr 2	Attr 3	Attr 4	Attr 5	Attr 6	Attr 7	Attr 8	Attr 9	Attr 10	Attr 11	Attr 12
0	2.75e-005	7.25e+000	-3	-1	-2	-2	-3	-2	-1	-1	-1	0	1	5	3
1	3.34e-006	8.08e+000	4	3	0	0	0	0	0	0	0	0	6	-1	-1
2	1.32e-005	8.15e+000	19	0	-2	-2	-2	-2	-1	-1	0	0	2	3	2
3	1.56e-006	6.34e+000	-1	0	-3	-2	-2	-2	-3	-2	0	0	2	3	1
4	2.79e-007	1.08e+001	-3	12	0	0	0	0	0	0	0	0	0	1	1
5	1.00e+000	1.77e+001	23	4	8	9	7	5	2	1	6	9	-7	-5	-7
6	4.61e-006	1.08e+001	-8	-5	0	0	0	0	0	-1	0	0	0	4	2
7	8.21e-008	8.34e+000	-2	-1	0	-1	-1	-1	0	0	-1	0	-1	0	0
8	3.08e-007	1.08e+001	-4	-4	1	0	0	-1	1	1	-1	0	0	3	2
9	1.06e-007	8.21e+000	0	3	-1	-1	0	0	0	0	0	0	6	0	0
10	3.26e-005	2.39e+001	19	0	2	4	3	2	-2	-1	8	0	1	-1	-1
11	2.58e-008	1.10e+001	0	0	0	0	0	0	0	0	0	0	2	-1	-1
12	4.00e-008	9.73e+000	0	3	4	2	3	3	4	3	0	0	2	-2	-2
13	1.87e-007	1.34e+001	-6	0	0	0	0	0	0	0	0	0	-1	-16	-11
14	2.53e-008	1.36e+001	-4	-5	1	0	0	0	1	1	0	0	-1	2	2
15	2.73e-008	1.28e+001	19	0	0	0	0	0	1	0	0	0	0	1	1
16	3.88e-003	1.68e+001	3	1	2	3	3	2	2	0	3	-2	-1	-4	-4
17	1.29e-003	1.53e+001	3	3	1	3	3	3	1	0	6	-2	-1	-9	-10
18	1.18e-006	1.50e+001	4	2	0	0	0	0	0	0	1	0	1	-13	-11
19	7.97e-006	1.69e+001	2	0	3	1	0	-1	-1	-1	-1	3	-12	-3	-5
20	4.81e-007	1.30e+001	1	0	1	2	3	2	1	1	2	-1	-1	-2	-2
21	6.15e-002	3.01e+001	3	1	12	11	7	3	0	0	5	9	-12	-4	-5
22	1.84e-008	2.94e+001	2	0	0	0	0	0	0	0	0	0	-1	-15	-11
23	3.17e-005	1.59e+001	19	1	3	4	3	3	1	1	5	7	-3	-2	-2
24	6.69e-006	1.54e+001	5	3	4	2	1	1	1	1	0	3	-2	-5	-7
25	1.00e-009	3.93e+001	-7	0	1	1	1	1	1	1	0	0	0	-2	-2
26	8.07e-009	5.59e+001	19	0	0	1	1	1	0	0	1	0	1	-1	-1
27	1.04e-008	6.24e+001	0	12	-1	-1	-1	-1	2	3	0	0	0	0	0
28	4.79e-008	2.32e+001	-5	-4	3	3	2	1	2	2	0	0	-5	2	2
29	1.45e-010	6.98e+001	3	0	1	1	1	1	1	0	0	0	0	-2	-2
30	3.12e-009	3.81e+001	0	0	-2	-2	-2	-1	-2	-1	0	0	2	2	2
31	1.27e-010	1.13e+002	-5	0	1	1	1	0	0	0	0	0	0	2	1
32	1.88e-010	1.63e+002	-7	-10	2	1	1	1	2	3	0	0	-2	1	3
33	5.19e-007	4.45e+001	19	0	2	3	2	0	-2	-1	8	0	1	-1	-1
34	1.91e-011	1.21e+002	0	0	1	0	0	0	0	1	0	0	-1	0	0
35	1.72e-009	7.92e+001	0	0	-2	-1	-1	-1	-1	-2	0	0	2	1	0
36	8.89e-011	3.52e+002	-2	-3	1	1	1	0	1	2	0	0	0	-1	-2

37	2.66e-011	1.38e+002	1	-1	-1	-1	-1	-1	0	0	0	0	1	1	2
38	2.02e-011	1.82e+002	0	0	3	3	2	1	0	0	1	0	0	0	1
39	3.04e-011	2.09e+002	-2	-3	0	0	0	0	0	0	-1	0	0	0	0
40	1.31e-012	5.48e+002	0	1	1	1	1	1	1	1	0	0	0	0	0
41	8.80e-011	5.13e+002	0	1	-1	-3	0	0	0	0	-4	0	1	0	0
42	1.18e-010	3.95e+002	-1	0	0	0	0	-1	0	0	0	0	0	1	0
43	3.49e-011	4.46e+002	0	-1	0	0	-1	0	0	0	0	0	0	2	1
44	4.23e-014	2.27e+003	-1	0	0	1	0	0	0	0	1	0	0	0	0
45	5.40e-011	1.76e+003	1	0	0	0	0	0	0	0	1	0	3	0	-1
46	8.51e-010	1.71e+003	-5	0	0	-2	0	0	0	0	-7	0	0	2	2
47	5.63e-011	1.21e+004	8	3	0	1	1	0	1	0	1	-1	0	-1	-2

Table 7.1. Relative class strength, class divergence and deviation of each attribute from overall distribution for each Autoclass cluster.

Finally, we have analyzed the Autoclass clusters according to the vector of probabilities of each instance to be member of a cluster to verify how good is the fit of identified classes to a multivariate normal distribution. The procedure of verification has been as follows. We have investigated the second probability of membership of each instance. We have calculated the percentage of instances in each cluster with probabilities greater than 25% in this second probability (this value has also been selected arbitrarily). The results are that none cluster presents a very overlapped probability distribution with other cluster. What we can say is that most clusters representing the same type of variable stars have probability distributions with any kind of overlapping. But there are exceptions. Following we show just some cases.

Cluster 0 and Cluster 6 representing OSARGS.

Cluster 0. This cluster contains OSARS. However, instances of this cluster do not have any probability to be members of cluster 6, the other cluster with this type of variable stars.

The probabilities point to other non-identified clusters: 3, 8 and 4 as the most relevant and in order of importance. However very few instances (a maximum of 3.5% for cluster 3) have relevant probabilities (more than 25%) of being membership on these clusters. The mixture of membership comes from neighbouring clusters with not very well defined frontiers.

Cluster 6. This is the other cluster that corresponds with OSARGS stars. We consider this other cluster moderately well separated because its instances have correspondence with clusters 3, 14, 7 and 18 in a small percentage (a maximum of 4.8% with cluster 3 with probabilities greater than 25%).

Both clusters have as one of the most relevant attributes the logarithm of the first frequency in relation to the global classification but both clusters do not share the range of values of this attribute.

So we can conclude that, although both clusters are formed by the same type of stars, the differences stated in point 4.5 makes them a different type of stars and need two different multivariate normal distributions to model them.

Cluster 16, 17, 18 and 20 representing eclipsing binary systems

Cluster 16. Cluster 16 only has an overlapping with cluster 20, another cluster with the same type of stars, but the percentage of instances is small. A 14.4% of instances has any probability of being member of cluster 20 but only a 0.8% with a percentage greater of 25%.

Cluster 17. Instances of this cluster have probabilities of being member of clusters 20 and 18. The 33.2% have any probability of being member of cluster 20 but only a 1.2% with a probability greater than 25%. The 17.4% have a probability of membership in cluster 18 but this percentage reduces to 1.7 % with a great probability.

Cluster 18. Cluster 18 is the cluster worse separated. It has overlapping with clusters 1, 22, 11, 26, 17 and 20 in order of importance. A 7% of instances has a percentage greater than 25% of being members of cluster 0.

Cluster 20. This cluster is related with clusters 17, 16 and 7. The greatest percentage, a 2.73%, of instances, have a probability of class membership of cluster 17.

The attributes log-1 and log-f2 make the main difference between cluster 16 and 17 because, it seems that the spurious detection of log-2 proportional to log-f1 is enough to separate them in two well differentiated clusters. So, although for Autoclass cluster 16 and 17 are perfectly separated, the separation is based on an attribute with erroneous values what makes this separation without meaning.

To know if these clusters maintain its composition we asked Autoclass to repeat clustering twice only with the instances that are part of these four clusters. Autoclass found more clusters (6 and 10) but there were not a mixture of instances in classes what means that Autoclass finds any type of structure in data and is able to make subtle distinctions among data. We also asked Autoclass to repeat clustering with 12 attributes without log-f2, the attribute that presents some spurious data, to know if these values were affecting clustering. This time Autoclass found 9 classes but they did not maintain their composition. New clusters present instances coming from the four original clusters.

The explanation to this is that Autoclass has been forced to find models with correlated attributes as strongly happens with attributes log-f1 and log-f2. It is logical that Autoclass finds these models more probable because they better describe the data distributions. At this point, it is problem of the domain expert to decide if the attribute log-f2 should be used or if data presenting erroneous

values in this attribute should be removed or filtered before clustering. Other possibility is not to correlate attributes that present this behaviour.

Cluster 19, 21 representing cepheid stars.

Cluster 19. This cluster corresponds with cepheid stars. Some instances have any probability of membership in clusters 24 (rrlyrae) and 21 (cepeids). A 13.2% of instances have any probability of being member of cluster 24 but only a 1.3% has a probability greater than 25%. The probabilities with cluster 21 are lower than with cluster 24, a 3.4% has any probability in cluster 21, and a 0.3 with a relevant probability.

Cluster 21. This other cluster containing cepheid stars has any overlapping with clusters 19 and 34. A 27.8 % of instances has a probability of being member of cluster 19 but only a 1.5% with a relevant percentage. Cluster 34 is a non identified cluster with very few weight in the classification. The percentage of membership is insignificant.

The most important attributes to differentiate these two clusters are log-af1h1-t, pd12 and log-crf10. They make think in two different distributions representing different objects than in two different distributions representing a single object that does not follow exactly a multivariate distribution.

Cluster 5, 23, 24 representing RRLyrae stars.

Cluster 5. This cluster has instances with a probability of membership to clusters 23 (24.2%) and 24 (5.1%). However the number of instances with a probability greater than 0.25% is about 1.2 % so we consider it a well separated cluster.

Cluster 23. This cluster has instances with a probability of membership to clusters 5, 24 and 29. A 31% of instances have a probability of belonging to cluster 5, and from this, a 5.6% with a percentage greater than 25%. A 10.2% of instances have a probability of belonging to cluster 24, but this percentage reduces to 1.2% with a relevant probability.

Cluster 24. This cluster has instances with a probability of being members of clusters 5 (10.5%), 29 (8.5%), 19 (7.7%) and 23 (3.8%). The major number of instances with a probability greater than 0.25% is a 2.8% in cluster 5.

Cluster 24 has more differences in values, but cluster 5 and cluster 23 look as being the same type of object. It seems that it is necessary the use of two probability distributions to define the type of stars.

Autoclass was asked to repeat twice the classification with the instances of these 3 clusters. Autoclass found 5 clusters in both runs with basically the same composition than in the previous one.

Analyzing 2D projection plots, one could conclude that 5, 23 and 24 are representing the same type of stars. The three clusters have basically the same mean values in all attributes but the correlation matrix presents different orientations in the ellipsoid formed by the sigma values. Instances that present more diffuses probabilities of being member of a class are located near of the mean values. See Figure 7.1.

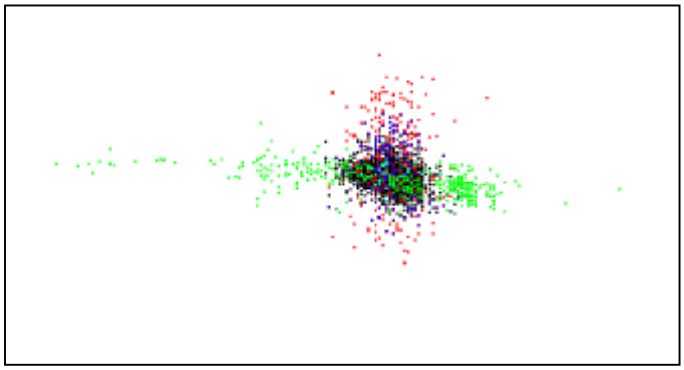


Figure 7.1, Detail of the location of instances of cluster 5 (red), 23(red) and 24(green). Instances of cluster 23 having a probability of being member of cluster 5 in blue, Instances of cluster 23 having a probability of being member of cluster 24 in cyan. The plot is in the log-f1 and V-I color index plane.

However, there are also subtle differences that figure 7.1 does not show. Cluster 23 presents log-f2 values with the greatest dispersion and cluster 24 show differences in mean values in the log-af1h3-t and log-af2h4-t attributes what lead to the three different distributions that Autoclass finds. So we conclude that these are the reasons why Autoclass needs three multivariate distributions to define this type of stars

Clusters 8, 14 and 28: Mira and semirregular stars.

Cluster 8. Cluster 8 has mainly an overlapping with cluster 0 (OSARGS), 14 and 28 in order of importance. A 33.3% of instances has any probability of being member of cluster 0 and a 8.7% with a percentage greater than 25%. The percentages with clusters 17 (22.3% - 3.4% relevant) and 18 (4.3% - 1.7% relevant) containing the same type of stars are minor.

Cluster 14. Instances of this cluster have probabilities of being member of clusters 32, 8, 27 and 28. The 18% have any probability of being member of cluster 32, cluster not identified, but with very few relevance. Although the global probability of instances of being member of cluster 8 is lower than with cluster 32, the relevant percentage increases to 4%.

Cluster 28. Instances of cluster 28 have probabilities in clusters 8 (28.0%), 14 (22.2%), 6 (11.5%) and 7 (5.7%). Greatest relevance is with cluster 8 (4.7%).

Cluster 8 and cluster 0 (OSARGS) have mean values very different in several attributes so overlapping has to be produced with outlier instances. Both clusters can be considered different distributions. Cluster 28 is a cluster with very few weight in the classification and with a value of a relative class strength very small: 4.79e-008 so it is predictable that this cluster could disappear in other run. Cluster 14 has also a value of relative class strength very small 2.53e-008. The major differences with cluster 8 comes from log-f1 attribute with mean and sigma values far enough to consider them different distributions.

Cluster 1, cluster unidentified.

Cluster 1. This cluster, the second in importance, shows how spurious data in attributes log-f1 and log-f2 affect to the classification. This cluster has not been identified due to these erroneous values in these attributes.

This cluster presents some overlapping (less than 3% the greatest value) with clusters 9, 12, 3, 11 and 18 in order of importance. But this overlapping is not important. These clusters have in common noise in attributes log-f1 and log-f2 what explains this mixture in some instances.

7.3 HMAC evaluation

In HMAC, there is not a model fitting. In HMAC each cluster is characterized applying a non parametric density estimator which can be useful when clusters deviate substantially from any parametric distribution. In a theoretical basis, this approach should give very good results, even better than parametric approaches since they are not constrained to fit data to a model that perhaps data do not follow exactly. However, now one can see that the results are not as good as one could expect.

For example, in the case of RR Lyrae stars, finally we have decided that this type of stars deviate from a multivariate normal distribution and this is the reason Autoclass needs three clusters to characterize them. HMAC should have been able to find just one cluster for this type of stars. And this is what has happened. HMAC has grouped the three clusters that Autoclass founds for this type of stars. But, HMAC has also grouped spurious data in the log-f1 attribute and find some difficulties in separate them from fundamental mode cepheids. So, one can also start to detect some problems in this approach.

In relation with eclipsing binary systems, HMAC found only one cluster that contains this type of stars as opposed to Autoclass that found four. But, in this case one can not be sure that this is good. The domain expert was able to see in one of these Autoclass clusters of eclipsing binary systems a subtle

distinction that could suggest a new classification scheme that had to be investigated. Perhaps this cluster, after a detailed study, does not conduct to a new classification, but HMAC eliminates any possibility of doing it. Clustering by density is canceling those small differences that could conduct to a new class discovery.

The characterization of Cepheids stars by HMAC was basically good. HMAC found first overtone pulsator Cepheids without any problem. First overtone pulsators appeared aggregated to RR Lyrae but they could be segregated in lower hierarchical levels.

Other clusters are more difficult to analyze because of the lack of information . For example, ellipsoidal stars cluster found by HMAC can only be compared to the same Autoclass cluster. Although, the core of these clusters is the same, both clusters aggregate surroundings instances that make them lightly different. In this case, there is not a clear criteria to decide which one is best. The same happens with BE stars. Autoclass and HMAC find a cluster in the same location but the size and shape of the clusters have small differences. If one wanted to venture to decide which one is better, visual analysis over the log-f1 vs V-I plots points to select HMAC cluster for ellipsoidal stars and Autoclass cluster for BE stars. But this is only a selection criteria based on the cluster shape in two attributes where the clusters appear sharply defined without having into account the rest of them.

Finally, OSARGS and Mira and Semiregular stars remain to be analyzed. Autoclass found a relationship between these types of stars and HMAC presented them joined. Both method have produced several clusters to characterize these type of stars what makes think in the difficulties in separating these types of stars. Attending to the identification done by the expert domain in Autoclass clustering, HMAC seems to be the approach working worse.

The additional information that HMAC supplies for analysis is the separability matrix. Separability values are not symmetric, this is, the separability values of two clusters can be different depending on which one is considered as reference to calculate the value. As, the analyzed level had 494 clusters, it is complicated to show this matrix. But, in general, the separability measure of clusters analyzed show good separabiliy values.

8. Conclusions and future work

In this master thesis, the focus of the study was to asses the validity of two unsupervised clustering algorithms based on very different approaches as a clustering tool for astrophysical data. Both have shown pros and cons.

1. About native implementation of both algorithms

Related to native C implementations, Autoclass is not totally viable to process the expected amount of data unless the problem described in section 4.3 be exactly identified and corrected. The hypothesis is that the problem is related to the use of duplicated instances to increase the dataset size in the experiments. EM algorithms, or variants like this, can not proceed if observations are very nearly colinear. EM breaks downs when the covariance matrix corresponding to one or more components becomes ill-conditioned (singular or nearly singular). This problem gets worse because of the measurement error specified for each attribute in the classification that treats each value as attr_value ± 0.05 (the error was specified equal for all attributes) making that the dataset have probably more similar instances than the expected. In fact, if this error is decreased to 0.0005 and the dataset is processed exactly in the same conditions than in section 4.5, the number of clusters increases from 48 to 58 showing the influence of this parameter.

HMAC is impractical as it is coded now because of the memory requirements (and bugs in code related with memory allocation) and the high processing time it needs, considering HMAC approach by its own, to reach a solution. In fact, it has not been able to process the main dataset without any code optimization.

However, parallelization seems to be possible in both algorithms to reduce processing time.

2. About number of clusters

HMAC results presents a limitation compared with Autoclass because it not provides exactly a final number of clusters as Autoclass does. This would not be so important if one could find in just one level all cluster with astrophysical meaning. But HMAC forces to choose clusters in several hierarchical levels so an important knowledge injection is required about what one wants to get. This problem seems to come from the different sizes, shapes and data points densities of the clusters. For Autoclass this is not a problem because each cluster is characterized by variables as the magnitude of the correlation or the relative sizes of the classes that constitutes the specific model parameters for each cluster. In fact, HMAC number of clusters and clusters have only been identified in comparison with the ones obtained by Autoclass.

3. About additional information supplied by the algorithms

Autoclass supplies a valuable information to analyze clustering results, relative entropy, class strength, attribute influence, covariance matrix,... HMAC gives separability values among clusters, a hierarchical structure that allows to establish relations between clusters,.. Depending on the context, one information is more interesting than the other.

4. About the efficiency finding new (synthetic) classes

Autoclass has shown high sensitivity detecting synthetic clusters formed with very few dispersed instances only with the condition that they follow a multivariate Gaussian distribution but HMAC does not. HMAC requires higher densities and closer distance between points to detect them as a cluster. Even so, HMAC also requires that small clusters be at a sufficient distance from prominent clusters to have enough separability from them.

5. About the ability to classify new instances

Autoclass has the ability to predict the classification of new instances and this presents another advantage since HMAC does not have it. This allows to have an additional tool to evaluate the goodness of clustering.

6. About data preprocessing

HMAC has not shown convincing results (to the author). The main problem found is stated in point 2 and in how it handles dispersed points (outliers). Perhaps a preprocessing step could have improved the results. HMAC uses a Gaussian kernel having a spherical covariance matrix with a standard deviation σ for all variables. This can be considered as using a “distance σ” as a parameter to merge points in each hierarchical level. This now makes think that data should have been normalized so all attributes have some standard mean and some standard deviation. However this preprocessing has not been performed. This work remains to be done.

About log-transformation, for both algorithms this preprocessing step has shown to be very suitable. Autoclass reduces processing time although it not improves results, HMAC improves classes discrimination considerably.

The OGLE dataset, the main dataset of this study, presents some evident problems with spurious data in some attributes. Although this situation seems to be not desirable and should be avoided, Autoclass is able to differentiate these data and form specific clusters containing these undesirable data whereas HMAC absorb them in clusters with astrophysical meaning. Perhaps using “missing values” instead of inventing correlated values in frequencies could be of interest to improve results. If datasets grow as it is expected, another solution could be to remove instances with spurious data to improve clustering. If the algorithm is stable, to reduce the dataset will not change clustering although it

will not classify all instances. Attribute selection could be another possibility. The influence analysis of noisy attributes could also conclude that these attributes could be removed from analysis without changing main clusters pattern. All these preprocessing steps, it is sure that conduce to a reduced number of clusters and a better astrophysical interpretation of them.

7. About astrophysical interpretation

Autoclass has proved extremely useful at identifying the classical variability types. In general, HMAC presents the same clustering pattern than Autoclass but interpretation is more complicated (at least for a non expert in the domain). In addition, outliers are a problem for HMAC because in this dataset many data are considered outliers that are not joint to a cluster until final steps of clustering. If using merging mechanisms of the algorithm, all outliers are added to the same cluster creating a very big cluster that losses its astrophysical meaning.

The tests described in this report prove that the parametric approach to clustering with Gaussian Mixtures in the OGLE variability context is adequate and correctly describes the main components of the point density distribution, even in cases where one attribute deviates strongly from gaussianity. The interpretability of gaussian components is straightforward and simple, the disadvantage being the necessity to fit several components (clusters) to describe a group that deviates from gaussianity. These components are not qualitatively different groups, and their analysis therefore distracts attention from the more interesting cases.

In the case of the need to fit several components to describe a class, HMAC has shown to be useful to merge clusters with the same meaning. In fact, HMAC is more effective to be used for merging previous clustering than for a complete clustering over the total large dataset.

8. About objective performance evaluation

Unsupervised clustering evaluation is still an open question. Although there are in the literature some proposed indices to carry out this task it seems they supply uneven results. Among them, the stability index could be an interesting indicator to incorporate to the ones used to evaluate algorithms in the CU7 context. It shows that the results are stable with respect to the sampling process and that the model proposed by some algorithm fits to the data or not. But this index only analyze the suitability of an algorithm but does not give an indicator to compare among algorithms. But this is an important index for the Gaia mission that will see its dataset increased along the time and the algorithm used must assure that the results are stable and clustering does not change drastically with the incorporation of new observations.

9. About other algorithms to test

Autoclass has supplied very interesting results even though the real clusters can not be modeled by a basic parametric distribution. Algorithms that can deal each cluster with a mixture of normals could supply better results.

There are in the literature different approaches to probability model selection to fit data that could also be interesting to try.

10. About the difficulties to carry out this work

Clustering seems to be a very interesting exploratory tool for astrophysical data. But unsupervised clustering algorithms does not supply a perfect solution that can be managed by anyone. Validation requires that clusters found are consistent with the prior knowledge one has about sample categories or data in the problem domain. If the analyst does not have that knowledge then he/she founds unable to decide if results are useful and to propose or find new solutions. Variable stars data is a complicated domain to handle.

9. Bibliography

[1] Data mining. Practical machine learning tools and techniques. Ian H.Witten & Eibe Frank
[2] http://www.esa.int/esaSC/120377_index_0_m.html
[3] The lognormal distribution in environmental applications. A.K.Singh, A.Sing, M. Engelardt. EPATechnology support center issue. EPA/600/R-97/006 Dec 1997.
[4] Recent advances in clustering: a brief survey. S.B. Kotsiantis, P.E. Pintelas
[5] Survey of clustering data mining techniques. Pavel Berkhin.Accrue Software. 2002
[6] Bayesian classification(Autoclasss): Theory and results. Peter Cheeseman, Jon Stutz.
[7] Bayesian classification theory. Peter Cheeseman, Robin Hanson, John Stutz.
[8] Bayesian classification with correlation and inheritance. Robin Hanson, John Stutz, Peter Cheeseman. Learning and Knowledge Acquisition.
[9] Tutorial on maximum likelihood estimation. In Jae Myung. Journal of Mathematical psychology 47 (2003) 90-100.
[10] A nonparametric statistical approach to clustering via mode identification. Jia Li, Surajit Ray, Bruce G. Lindsay. Journal of Machine Learning Research 8 (2007) 1687-1723.
[11] P-Autoclass: Scalable Parallel Clustering for Mining Large Data Sets. Clara Pizzuti and Domenico Talis. IEEE Transactions on knowledge and Data Engineering. Vol 15. No3, May/June 2003.
[12] Bayesian clustering with AutoClass explicitly recognises uncertainties in landscape classification. J. Angus Webb, Nicholas R. Bond, Stephen R. Wealands, Ralph Mac Nally, Gerry P. Quinn. Ecography 30: 526-536, 2007
Peter A. Vesk and Michael R. Grace.
[13] Clustering Based on a Multi-layer Mixture Model. Jia Li. Journal of Computational and Graphical Statistics (14)3:547-568
[14] Model-based evaluation of clustering validation measures. Marcel Brun, C. Sima, J. Hua, J. Lowey, B. Carroll, E. Suh, E. R. Dougherty. Pattern Recognition 40 (2007) 807-824
[15] How to evaluate clustering techniques. Daniel Delling, M. Gaertler, R. Görke, Z. Nikoloski, D. Wagner.
[16] Performance evaluation of some clustering algorithms and validity indices. Ujjwal Maulik. IEEE transactions on pattern analysis and machine intelligence, Vol 24, No 12. Dec 2002
[17] An objective evaluation criterion for clustering. Arindam Banerjee, J. Langford.
[18] A comparison of clustering techniques in aspect mining. G. Serban and G. S. Moldovan. 2006

[19] How many clusters? Which clustering method? Answers via model-based cluster analysis. C. Fraley and A.E.Raftery, Tecnical report No. 329
[20] Evaluation of Hierarchical clustering algorithms for documents datasets. Ying Zhao, G. Karypis. 2002.
[21] Clustering combination method. Yuntao Qian; Suen, C.Y. Pattern Recognition, 2000. Proceedings. 15th International Conference on Volume 2, Issue , 2000 Page(s):732 - 735 vol.2
[22] Toward a statistical theory of clustering. U. von Luxburg and S. Ben-David. Presented at the PASCAL workshop on clustering, London. Technical report (2005)
[23] Stability-based model selection. T. Lange, M.L. Braun, V.Roth, J. Buhmann.
In Advances in Neural Information Processing Systems (2003).
[24] A stability based method for discovering structure in clustered data.A. Ben-Hur, A.Elisseeff, I.Guyon. Pac Symp Biocomput. 2002 ; :6-17
[25] Details of the Adjusted Rand index and Clustering algorithms. Supplement to the paper “An empirical study on Principal Component Analysis for clustering gene expression data” (to appear in Bioinformatics) Ka Yee Yeung, Walter L. Ruzzo. May 3, 2001

Astronomy & astrophysics
[26] Sarro, L., Debosscher, J., López, M., Aerts, C., 2008, A&A, ADS Link
[27] Sarro, L.M., Sánchez-Fernández, C., Giménez, A., 2006, A&A, 446, 395, ADS Link
[28] Soszynski, I., Udalski, A., Szymanski, M., et al., 2003, VizieR On-line Data Catalog: J/other/AcA/53.93. Originally published in: 2003AcA....53...93S, 50, 5301, ADS Link
[29] Soszynski, I., Udalski, A., Kubiak, M., et al., 2004, Acta Astronómica, 54, 129, ADS Link
[30] Udalski, A., Soszynski, I., Szymanski, M., et al., 1999, Acta Astronómica, 49, 223, ADS Link