Available online at www.sciencedirect.com

ScienceDirect

Pattern Recognition Letters 27 (2006) 1590–1605

Pattern Recognition Letters

www.elsevier.com/locate/patrec

# Wavelet based approach to cluster analysis. Application on low dimensional data sets

Xavier Otazu [a,*], Oriol Pujol [a,b]

[a] *Computer Vision Center, Campus UAB, Research Division, Cerdanyola del Vallès, 08193 Barcelona, Spain*
[b] *Dept. Matemàtica Aplicada i Anàlisi. Universitat de Barcelona, 08007 Barcelona, Spain*

Received 26 April 2004; received in revised form 5 March 2006
Available online 3 May 2006

Communicated by M. Kamel

## Abstract

In this paper, we present a wavelet based approach which tries to automatically find the number of clusters present in a data set, along with their position and statistical properties. The only information supplied to the method is the data set to analyze and a confidence level parameter. Most of the usual methods for cluster analysis and unsupervised classification do not automatically determine the number of clusters present in our data. Thus, the human operator has to supply the method with an a priori number of clusters which the algorithm is expected to find. This fact leads to a difficult interpretation of the resulting clusters. In this paper we also show a practical algorithm to implement this method on low dimensional data sets.

*Keywords:* Wavelet Transforms; Image classification; Clustering methods

## 1. Introduction

Clustering techniques are widely used in pattern recognition to obtain information from multidimensional data sets when the problem domain is totally unknown or the number of classes cannot be defined beforehand. In this sense, they are opposed to semi-supervised or fully supervised techniques, in which prior knowledge is exploited for training purposes in order to obtain meaningful data sets or high performance classification procedures.

In general, clustering is defined as an unsupervised machine learning process in which data is grouped according to a notion of proximity. Therefore, each cluster is defined as a set of data points that are "close" to each other. Usually, clustering techniques are roughly divided in: partitioning techniques (based on the iterative refinement of a first random partition) (Kaufman and Rosseeuw, 1990), mixture model based methods (related to density estimation processes) (McLachlan and Basford, 1988), hierarchical clustering (hierarchical relationship among data) (Hartigan, 1975; Holschneider and Tchamitchian, 1990; Kaiser, 1994; Kaufman and Rosseeuw, 1990; Kohonen, 1988).

However, most of the methods in the former taxonomy do not produce a suitable estimation of the number of output clusters by themselves and it has to be provided as an input parameter. Regarding this last issue, several approaches have been proposed for determining the number of clusters: cross-validation (Smyth, 1996), penalized likelihood estimation (Sugiyama and Ogawa, 2001), permutation tests, resampling and finding the knee of an error curve (Tibshirani et al., 2003). But the complexity of most of these methods is very high since they demand multiple runs of the clustering algorithm.

As a result of the drawbacks of classical techniques, another important line of work in clustering was born:

[*] Corresponding author. Tel.: +34 935813036; fax: +34 935811670.
*E-mail addresses:* xotazu@cvc.uab.es (X. Otazu), oriol@cvc.uab.es (O. Pujol).

0167-8655/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2006.03.006

robust statistics techniques. In particular, robust clustering differs from the classical approaches in the fact that they try to solve the following standard problems: (1) robustness to the initialization (initial guesses and priors); (2) robust to cluster volumes and (3) robust in front of noise and outliers. However, few are the algorithms that are able to give answers to all those problems. Recently, Yang and Wu (2004) proposed a robust approach, similarity based robust clustering (SCM), that was able to cope with those problems as well as outperform other well-known and high-performance clustering algorithms such as fuzzy c-means or possibilistic c-means.

In this paper we present a wavelet-based approach to perform cluster analysis on multidimensional data sets, pursuing to define a procedure or algorithm which only works with the data to analyze and a confidence parameter. We compare our method with classical K-MEANS, hierarchical clustering with automatic cluster determination and the aforementioned similarity based clustering, SCM. We provide synthetic experiments, and application to color real images.

## 2. Related works and motivation

This section is devoted to give a brief overview of the techniques we will use to compare our method with. We will compare the results with three methods: the classical K-MEANS clustering; hierarchical clustering with automatic determination of the number of clusters using evaluation graphs; and finally, the similarity-based robust clustering. K-MEANS has been chosen because it is a standard well-known and well-studied method that allows basic comparison. Hierarchical clustering with graph evaluation has been chosen as a standard representative of agglomerative techniques. In particular we have also included the automatic search of the number of clusters, which will be further discussed in the next subsection. Finally, a robust all-purpose method is also compared. This method has been chosen as a representative of the robust methods for its proved effectiveness. Further details of the method are given in the next subsections. The last subsection briefly introduces the motivation of this work.

### *2.1. Finding the knee of an evaluation graph*

Analysis of evaluation graphs is a classical technique for finding the number of clusters in unsupervised learning. An evaluation graph consist of a plot displaying the number of clusters vs. an evaluation metric. Several methods are used for this analysis: finding the largest magnitude difference between two points, the largest ratio difference, the first data point with a second derivative above some threshold,... However, recently, a new method, *L method*, has been proposed that is considered to obtain a good approximation of the number of clusters.

This method splits the evaluation graph plot in two linear regression models ($L_c$,$R_c$). The crossing of both lines determines the number of clusters. The goal is to minimize the following expression:

$$\mathrm{RMSE}_c = \frac{c-1}{b-1}\mathrm{RMSE}(L_c) + \frac{b-c}{b-1}\mathrm{RMSE}(R_c),$$

where RMSE stands for *root mean squared error*, $b$ is the maximum number of clusters, and $c$ is the partition point. $L_c$ is the model that considers all the measures of the clusters on the left side of the partition point $c$. $R_c$ does the same on the right side of the partition point.

This is a general method without parameters that is specially well suited to be used in conjunction with hierarchical clustering, since the evaluation measure can be the same as the one used for creating the hierarchical dendrogram. However, the evaluation graph resulting from a hierarchical clustering is as large as the original data set, since it treats every single data point as a cluster. In this situation the representability of the knee of the L shaped curve is very low. In (Salvador and Chan, 2004), they propose an iterative refinement of the L-method by bounding the upper number of clusters to $2 \times c$ at each iteration, until the method converges. Hierarchical clustering in conjunction with the L-method (hereafter, HIER) is used for comparison purposes with the new method we present in this work.

### *2.2. Similarity-based robust clustering*

Similarity-based robust clustering (hereafter, ROBUST) is a newly proposed but effective technique capable of solving most problems in cluster analysis (Yang and Wu, 2004). In particular, it solves the initialization and volume issues by means of a self-organizing technique of the data.

The method is divided in two steps: The first step is the *correlation comparison algorithm (CCA)*. This algorithm is a previous step to the main clustering core. In this step the scale of clustering is fixed automatically by exhaustively calculating the correlation between $\widetilde{J}(x_k)_{\gamma_m}$ and $\widetilde{J}(x_k)_{\gamma_{m+1}}$ where

$$\widetilde{J}(x_k)_{\gamma_m} = \sum_{j=1}^{n}\left(\exp - \frac{\|x_j - x_k\|^2}{\beta}\right)^{\gamma_m}, \quad k = 1,\ldots,n$$

for $\gamma_m = 2m, m = 1,2,3,\ldots$ If the correlation of this density based estimator is higher than the 99%, we consider to have a good approximation of the clustering scale. This consideration is also proposed in (Yang and Wu, 2004). The second step is *similarity clustering algorithm (SCA)*. In this step, the goal is to find the centers $z_i$ that maximizes the similarity function $J_s(z)$

$$J_s(z) = \sum_{i=1}^{c}\sum_{j=1}^{n}\left(\mathrm{e}^{-\frac{\|x_j - z_i\|^2}{\beta}}\right)^{\gamma}, \quad (1)$$

where $n$ is the number of data points and $c$ is the number of centers. The parameter $\beta$ is defined as the sample variance and $\gamma$ is the resulting scale obtained in the former step. The initial data and centers are considered the same and the scheme is embedded in a maximizing framework, such as gradient descent of fixed point of the derivative of the similarity functional.

In this sense, this approach is a self-organizing technique, whose output is a set of cluster centers around which the input data points are grouped. SCA is the iterative procedure that allows each center to converge according to the similarity measure. The more similar the data is, the more centers converge to the same point. The number of final clusters is not known a priori, but the algorithm convergence determines this number.

## 3. Wavelet transform

Multiresolution analysis based on wavelet theory introduces the concept of details between successive levels of scale or resolution (Mallat, 1998; Chui, 1992; Daubechies, 1992; Kaiser, 1994; Meyer, 1993; Vetterli and Kovacevic, 1995). Wavelet decomposition is increasingly being used in many Data Processing topics (Starck and Pantin, 1996; Rué and Bijaoui, 1996) and it is based on the decomposition of the data set into multiple channels according to their local frequency content. Wavelet Transform decomposes data sets into a number of new sets, each one with distinct frequential information.

The wavelet representation is an intermediate representation between the Fourier and the spatial representation. The Fourier Transform represents the global frequency content of the image, but it provides no information on the spatial localization of these frequencies. However, the Wavelet Transform gives simultaneous information on both the local frequency content and the spatial distribution of these frequencies. Since in Fourier space the base functions are sinusoidal, they extend throughout space and do not have spatial concentration. A windowed Fourier transform can be performed using windowed sinusoids, but the size of these windows is the same, regardless of the frequency which fills it. On the other hand, wavelets are concentrated around a central point and so, as the windowed Fourier transform, they have a high degree of spatial localization, but the degree of concentration depends on the frequency content of this wavelet function. High frequency wavelets are narrower than lower frequency ones, and thus provide a sort of adaptive base functions, suggesting a zooming tool.

### 3.1. Theory

We shortly present some outlines in wavelet transform. We present one-dimensional formulas which can be easily extended to the $n$-dimensional case.

Given a signal $f(t)$, we can decompose it in the wavelet space as

$$W_{m,n}(f) = \int_{-\infty}^{\infty} f(t)\psi_{m,n}(t)\,\mathrm{d}t, \tag{2}$$

where $\psi_{m,n}(t)$ are the wavelet basis functions defined by the $m$ and $n$ parameters. The $\psi_{m,n}(t)$ basis functions are derived from a function $\psi(t)$ called *Mother Wavelet* which are defined as

$$\psi_{m,n}(t) = 2^{m/2}\psi(2^m t - n). \tag{3}$$

Using (3) and other constraints we obtain an orthonormal wavelet basis (Daubechies, 1992). Parameter $m$ stretches or compresses the *Mother Wavelet*, leading to a narrower or broader new function. Parameter $n$ translates the *Mother Wavelet* along $t$ space. Therefore, all the basis functions $\psi_{m,n}(t)$ have the same profile, i.e., the Mother Wavelet $\psi(t)$ profile, but dilated and translated according to parameters $m$ and $n$ respectively.

The inverse discrete wavelet transform is given by the reconstruction formula:

$$f(t) = \sum_m \sum_n W_{m,n}(f)\psi_{m,n}(t). \tag{4}$$

In summary, the wavelet transform describes a transition or an abrupt change at a given scale in the signal.

### 3.2. The à trous algorithm

In the discrete case, many of the wavelet transform algorithms are not shift-invariant, which can compromise image decomposition. In this work we need a shift-invariant discrete wavelet decomposition for $d$-dimensional data sets.

We can use the discrete wavelet transform known as *à trous* ('with holes') algorithm (Holschneider and Tchamitchian, 1990) to decompose our data into wavelet planes. Given a data set $p$ we construct the sequence of approximations:

$$F_1(p) = p_1, \quad F_2(p) = p_2, \quad F_3(p) = p_3, \ldots \tag{5}$$

To construct the sequence, this algorithm performs successive convolutions with a filter obtained from an auxiliary function named scaling function (ref. for further details). We use a scaling function which has a $B_3$ cubic spline profile. In the case of a one-dimensional data set, the use of a $B_3$ cubic spline leads to a convolution with a mask made by five elements being scaled up by 16: $(1,4,6,4,1)$. In the case of a two-dimensional data set (an image), the convolution mask is Starck and Murtagh (1994):

$$\frac{1}{256}\begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix} \tag{6}$$

and generalization to $d$-dimensional data sets is immediate. As stated above, wavelet planes are computed as the difference between two consecutive approximations $p_{l-1}$ and $p_l$. Letting $\omega_l = p_{l-1} - p_l$ $(l = 1,\ldots,n_\omega)$, with $p_0 = p$, we can write the reconstruction formula:

$$p = \sum_{l=1}^{n_\omega} \omega_l + p_r. \tag{7}$$

In this representation, data sets $p_l$ ($l = 1, \ldots, n_\omega$) are versions of the original set at increasing scales (decreasing resolution levels), are the multiresolution wavelet planes and $p_r$ is a residual set. Parameter $n_\omega$ is the number of wavelet planes. In our case, we use a dyadic decomposition scheme, hence, the original set $p_0$ has double the resolution of $p_1$, $p_1$ double resolution than $p_2$ and so on. If the resolution of $p_0$ is, for example, $10u$ (being $u$ a spatial unity of measure), then the resolution of $p_1$ is $20u$, the resolution of $p_2$ is $40u$ and so on. However, note that all consecutive approximations (and wavelet planes) using the à *trous* algorithm have the same number of data points as the original data set (working with images, we obtain the same number of pixels), given that the à *trous* algorithm is an oversampled transform (Vetterli and Kovacevic, 1995). The usual Wavelet Transform decomposition schemes performs a decimation on the data between consecutive planes, so every wavelet plane contains fewer data than the previous plane. In the à *trous* algorithm this decimation step is not performed, so the amount of data is the same for every wavelet plane, therefore every wavelet plane obtained by the à *trous* algorithm has the same amount of data as the original data set. This restricts the use of this particular wavelet approach for applications such as data compression.

## 4. Wavelet based model for cluster analysis

Since wavelet transform isolates features with different frequency content, it can be useful to detect, discriminate and localize clusters in multidimensional space.

Take a multidimensional data set, and place them in this space as defined by a multidimensional density distribution. If this distribution is Gaussian-like, clusters are located where point density is higher. Since the histogram can be used to estimate the density distribution, it is clear that clusters, i.e., multidimensional centroids, are located around the highest histogram values. Given a multidimensional histogram of a cluster, we can treat this histogram as a function. Since any function can be interpreted from a frequential point of view, we can see the relation between clusters location and histogram frequency content: clusters concentrated around a central point are high frequency features in the multidimensional histogram, and wide spreaded clusters are low frequency features. When performing a wavelet transform of the multidimensional histogram, these contributions are isolated in several wavelet planes, and this isolation process is guided by the frequential content (i.e., statistical properties) of clusters in the multidimensional histogram.

Hence, our goal is to find clusters in the wavelet space instead of finding them in the multidimensional space, as done by many cluster analysis algorithms. In this way, the multidimensional space is decomposed into several new spaces with their own statistical, i.e., frequential, properties. To detect and locate clusters, it is necessary to obtain local maximums in every histogram wavelet plane. A local maximum in a wavelet plane reveals the presence of a feature, related to the frequency associated with this plane, which may be linked to a cluster, and thus indicates the center of a candidate cluster. The wavelet plane where the maximum is found is associated with the statistical properties of this cluster. Once the contribution of the cluster in this wavelet plane has been isolated, we can directly measure its position, extension and profile, i.e., its statistical properties (mean position, covariance matrix, etc.). Therefore, wavelet decomposition of a multidimensional histogram can be useful for cluster detection, localization, and approximate estimation of its statistical properties.

The automatic detection of clusters from the multidimensional histogram has been studied by Letts (1978), who defined clusters as local peaks of the histogram. This approach is too conservative, because in an extreme situation a slight change in the number of pixels in a bin of the histogram can determine a local maximum or not. In some way, our approach can be seen as an improvement of this method, since we try to find the embedded subclusters that does not manifest as peaks.

### 4.1. Selection of candidates

Local maximums of the histogram wavelet decomposition are cluster candidates, but it is necessary to discriminate false clusters from the true ones.

For example, take a multidimensional data set defined by a Gaussian distribution function, contaminate it with white Gaussian noise, and calculate its multidimensional histogram. In the wavelet planes obtained from the decomposition of this histogram, we have coefficients which are not contributions from the true cluster but from the histogram quantization noise. These last coefficients may create 'false alarms' during the cluster detection process.

Thus, we need to detect and establish what coefficients are related to a true cluster and what are false alarms. In a wavelet decomposition, contribution of a signal feature is present in several wavelet planes, and its position is clearly linked to the true position of this feature. This approach has already been used in (Starck et al., 1998) to perform object detection. In contrast, coefficients due to noise are not associated and cannot be present in several wavelet planes. Hence, to obtain a true, significant feature, we have to look for correlated coefficients through the wavelet planes. Therefore, to isolate true clusters from false alarms, we have to look for these correlated wavelet coefficients.

The size of histogram bins, i.e., histogram resolution in the $d$-dimensional space, is usually an important issue in histogram-based procedures. High histogram resolution, i.e., many histogram bins, defines a sparse histogram with poor populated bins. On the other side, a small resolution histogram, i.e., few histogram bins, implies highly populated bins and a smooth histogram but a small number of bins. If we take a high resolution histogram, bins will be very poorly populated, which will introduce noise on the histogram distribution and many local maximum wavelet coefficients will not be associated to a cluster. On the other side, low resolution histograms produces highly populated histogram bins, but resolution of features in the $d$-dimensional space is compromised. Some prior knowledge about data distribution would be an advantage, but if we do not know this information an optimum point between these two situations has to be adopted.

In order to circumvent this problem we describe in the following section a strategy to calculate the resolution at which our histogram contains valuable information. As shown in the following section, it is equivalent to the estimation of a particular wavelet plane $\omega_{l_0}$, and also equivalent to the estimation of an optimum size for the size of histogram bins.

To find the clusters present in our data, we define the following algorithm:

- Obtain the $d$-dimensional histogram $h(\boldsymbol{p})$ of the data set $\boldsymbol{p} = \{p_j(x_1,\ldots,x_d), j = 1,\ldots,n_p\}$, being $n_p$ the number of data points.
- Perform the wavelet transform of the histogram $h(\boldsymbol{p})$, obtaining the wavelet planes $\omega_i(x_1,\ldots,x_d)$, $i = 1,\ldots,n_\omega$.
- Detect local maximums $C_{i,l}$ on every wavelet plane $\omega_l$, $l = l_0,\ldots,n_\omega$, being $l_0$ the initial wavelet plane.
- For every maximum $C_{i,l}$:
  - Given a wavelet coefficient $C_{i,l}$ which is a local maximum located in the wavelet plane $\omega_l$ $(c_1,\ldots,c_d)$, we look for local maximums $C_{i',l-1}$ and $C_{i'',l+1}$ in a window centered in the same position $(c_1,\ldots,c_d)$ of the $C_{i,l}$ maximum but in the wavelet planes $\omega_{l-1}$ and $\omega_{l+1}$, respectively.
  - If maximums $C_{i',l-1}$ and $C_{i'',l+1}$ are found at both $\omega_{l-1}$ and $\omega_{l+1}$ and the value of the coefficient $C_{i,l}$ is higher than both $C_{i',l-1}$ and $C_{i'',l+1}$, then we take $C_{i,l}$ as a real cluster. Otherwise $C_{i,l}$ is rejected as cluster.

Hereafter, this algorithm is called WAVCLUS.

We want to stress that this method is designed to work with multidimensional histograms that present an ellipsoidal profile with Gaussian-like density distributions. That is, it supposes that multidimensional clusters are distributed around a central point, and that this density decreases when distance from the central point increases.

The estimation of the initial $\omega_{l_0}$ wavelet plane from which we start the detection of local maximums is explained in the following section.

## 5. Optimum histogram resolution

The size of bins in which we discretize the $d$-dimensional feature space is an important issue to take into account when constructing a histogram. Narrow bins produce a high resolution histogram, but they are poorly populated and histogram becomes noisy. On the opposite, wide bins produce smooth histograms, but resolution is poor. Thus, a criteria has to be defined in order to find an optimum size for the bins.

On a clustering task, one of the concepts that have to defined is the minimum resolution at which data is considered to be statistically significant. For example, clustering techniques based on hierarchical methods define the resolution of data at several levels. When choosing one of the hierarchical levels, we are fixing the analysis resolution level of our data. For example, in the K-MEANS method, we fix a priori the number of clusters. If this number is small the obtained clusters will be large clusters that describe approximate general features. On the opposite, if the number of clusters is high, the obtained clusters will be smaller clusters.

One of the existing methods to analyze the optimum resolution of the distribution of our data in the $d$-dimensional feature space are the Parzen windows. The estimation of densities using Parzen Windows is performed counting inside a window the number of data points in the $d$-dimensional space. These windows may be defined in a very general way, which may allow to perform some kind of interpolation when using, for example, smooth Gaussian-like windows. The characteristic size of these windows define the resolution at which we estimate the density of our data set in the $d$-dimensional feature space.

Using the variable resolution concept to describe the density of our data into the $d$-dimensional feature space, Yang and Wu (2004) describe a technique to estimate the optimum resolution at which to describe our data set. They use a general kernel to smooth data distribution at different resolutions, and they perform a correlation between consecutive degrees of smoothing. When correlation value becomes higher than a predefined $c_{\min}$ value or when the correlation increment between consecutive degrees of smoothing is negative, then they take that degree of smoothing, i.e., the resolution of the smoothing kernel, as the optimum resolution to describe data density.

The WAVCLUS method also allows us to estimate the optimum resolution to describe our data set. When using the à *trous* algorithm during the multiresolution wavelet decomposition process, we smooth the original data using kernels of increasing size. We obtain a set of approximations $p_l$ of our original data set at decreasing resolutions. The differences between these consecutive approximations $p_l$ are the several wavelet planes $\omega_l$. Similarly to the previously described (Yang and Wu, 2004) method, we may calculate the correlation between consecutive approximations. Looking at these correlation values we may estimate the optimum resolution, i.e., the optimum $\omega_{l_0}$ wavelet plane, at which to describe our data.

Hence, we may define the following procedure in order to find the $\omega_{l_0}$ wavelet plane which corresponds to the optimum resolution to describe our data set:

- Construct the $d$-dimensional histogram using a small enough $s_b$ size for the bins.
- Obtain approximations $p_l$, $l = 1, \ldots, n_\omega$ and the wavelet decomposition $\omega_l = p_{l-1} - p_l$ of the histogram.
- Take $l = n_\omega$.
- Calculate the correlation $c_l$ between consecutive approximations $p_l$ and $p_l - 1$.
- If $c_l > c_{\max}$ then define $l_0 = l$ and finish, else decrement $l$ and goto previous step.

We note that we start the correlation analysis with the smoother approximation, i.e., the $p_{n_\omega}$ approximation. As Yang and Wu (2004) show in their work, this series of correlation values has to be interpreted from the lower resolution approximations of our data up to the more similar to the original data set. Thus, to find the optimum $l_0$ value we start taking $l = n_\omega$ and comparing $p_l$ with $p_{l-1}$, $p_{l-1}$ with $p_{l-2}$, and so on up to $p_0$, which is the original data set.

The obtained $\omega_{l_0}$ wavelet plane is the one with the optimum resolution to describe our data density. It implies that the clusters probably present in our data set are greater than this resolution size. Therefore, WAVCLUS method has to search local maximums at lower resolutions, i.e., at wavelet planes $\omega_l$ with $l > l_0$.

This procedure implies the size $s_b$ of the bins to construct the histogram is not a critical parameter for the WAVCLUS algorithm (Fig. 1). We just have to take a small enough value for $s_b$ in order to avoid collapse important features of the density distribution inside a single discrete bin. For example, take we chose $s_b = 1$ and that we obtain $l_0 = 3$. If we take $s_b = 2$ we will obtain $l_0 = 2$, and if we take $s_b = 3$ we will obtain $l_0 = 1$. The explanation is that since we take a greater size of the bins when incrementing $s_b$, the features becomes smaller in the new histogram and they appear in a lower $l_0$ plane. In this particular example, $s_b = 3$ is the maximum size for the histogram because we will obtain $l_0 = 1$, which is the lowest wavelet plane.

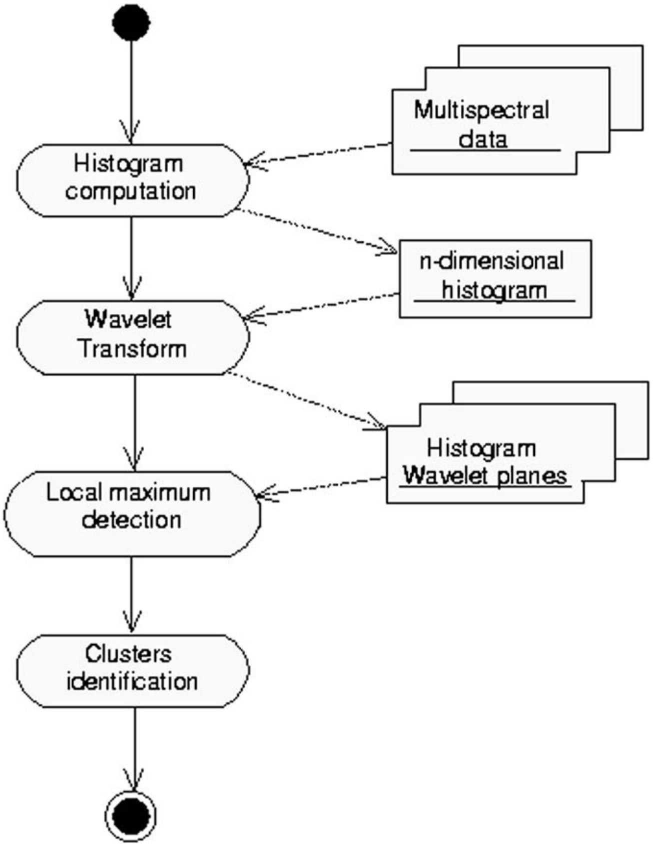

Fig. 1. UML activity diagram for the WAVCLUS algorithm.

We want to note that the $c_{\min}$ value is the only free parameter of the WAVCLUS method that the user has to define prior to the clustering process. In fact, $s_b$ has also to be defined but, as explained, the user has just to be careful to take a small enough value in order to avoid discretizing too much the data set.

## 6. A practical application on low dimensional data sets

In the following sections we present the results obtained when implementing the described WAVCLUS algorithm. The algorithm used to perform the wavelet decomposition is the à *trous* algorithm, using the filter in (6).

When implementing the above algorithm, some practical considerations on the implementation of the $d$-dimensional wavelet transform should be taken into account. When using the à *trous* algorithm to perform the wavelet transform, each wavelet plane is described by $(n_b)^d$ values, where $d$ is the number of dimensions in our $d$-dimensional data set, and $n_b$ is the number of bins used to compute the histogram (usually $n_b < 256$). When $d > 3$, the $(n_b)^d$ data size is too large to be managed by real computers, and some strategies have to be used in order to reduce the amount of data. A partial solution could be reducing the number of bins, for example using other wavelet transform algorithms that includes decimation. A principal component analysis to reduce the dimensionality of our $d$-dimensional data set prior to histogram calculation could be also very useful. Another solution would be to work only with the non-zero values of the $d$-dimensional histogram. Several computer considerations on the implementation of the algorithm are discussed in Section 10.

## 7. Simulated data

### 7.1. Data generation

To test the behavior and accuracy of the WAVCLUS algorithm, several synthetic data sets were created and classified by WAVCLUS, K-MEANS, ROBUST and HIER methods. Each data set contains several clusters which are created as Gaussian distributions in the multidimensional space, with distinct location and standard deviation. Several dimensions were used for the $d$-dimensional space, being $d = 1$, 2, 3. Within each data set, we created three clusters with different statistical properties to obtain clusters highly influenced by neighbor clusters, i.e., clusters which share a range of the $d$-dimensional histogram and can be considered as clusters embedded into other ones. Every cluster has been formed by a different number of points to measure the importance of cluster population in the detection and estimation of cluster statistical properties by the used clustering algorithms. For example, we created a highly populated cluster (made by 90% of total number of points) which presents a wide numeric range in the several dimensions. Close to this primary cluster, we created a second one (9% of points) with a different numeric range, and finally a marginal cluster with very few points (only 1%) but highly concentrated. None of them, except the highly populated one, can be located by the Letts local maximum histogram method described above, since none of them is a local maximum.

### 7.2. Accuracy estimation

If we know to what cluster a data point has to be assigned, the usual procedure is to construct a confusion matrix and calculate the accuracy of the classification method obtaining the percentile value of correct assignments. The $\kappa$ index is a good indicator of classification accuracy. From the confusion matrix, it can be obtained as

$$\kappa = \frac{n_p \sum_k x_{kk} - \sum_k x_{k+}x_{+k}}{n_p^2 - \sum_k x_{k+}x_{+k}}, \tag{8}$$

where $x_{ij}$ is the number of points classified as class $i$ but really belonging to class $j$, $x_{i+} = \sum_j x_{ij}$ (sum of all columns in $i$th row), $x_{+j} = \sum_i x_{ij}$ (sum of all rows in $j$th column), and $n_p$ the number of points in the $d$-dimensional data set. The closer to 1 is this number, the more accurate is our classification. If we performed a random classification using $N$ classes, we would obtain percentile indexes of correct classification for each class which would be around $100/N$%. For example, if we had only two classes, in a purely random classification we would obtain 50% of correctly classified pixels, but in this situation $\kappa = 0$. This value tells that our classification is completely random. It shows that $\kappa$ is a better indicator for classification accuracy of methods than simple percentile values.

### 7.3. Results

Beginning with a simple one-dimensional histogram, we created a data set with three clusters. The data set contained 1,048,576 points (1024 × 1024), and the numeric values ranged from 0 to 32, with a bin amplitude for the histogram equal to one numeric value (thus obtaining an histogram with 32 bins). The histogram obtained presents a Gaussian profile with slightly modified wings owing to secondary clusters (Fig. 2). In the first column of Table 4, we show their statistical properties and the percentile number of pixels used to create this cluster. We decomposed the histogram into four wavelet planes, i.e., $n_\omega = 4$. Fig. 2 shows the histogram and its wavelet decomposition. We can see that around $x = 5$, there is a clear peak in the first and second wavelet planes, $\omega_1$ and $\omega_2$ respectively. Around $x = 25$, there are also two peaks at the second and third wavelet planes, which indicates that this cluster candidate is a broader cluster than the one at $x = 5$. The central cluster at $x = 15$ is clearly visible at the two last wavelet planes and at the residual one.

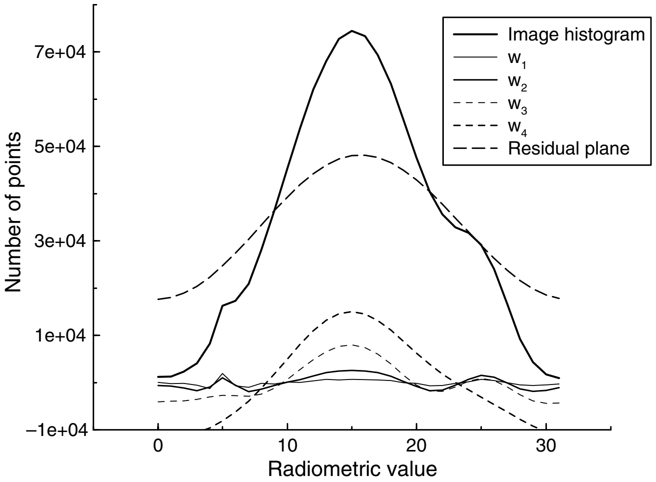

Fig. 2. Histograms of original image and the several wavelet planes $w_i$ obtained from the wavelet decomposition of the initial histogram.

As explained in previous section, in order to estimate the initial $l_0$ wavelet plane to look for clusters, we obtained the correlation between the several approximation data sets $p_l$. We show these values in Table 1. For all synthetic examples in this work, we take $c_{\min} = 0.98$.

In this case, the $p_4/p_5$ correlation value, e.g., 0.967, is lower than the desired 0.98 value. The following correlation value $p_3/p_4$, e.g., 0.974 is higher than the previous one, but lower than 0.98. The $p_2/p_3$ value is 0.996, which is higher than 0.98, hence we define $l_0 = 2$. It means that we search for clusters, i.e., local maximums in the wavelet planes, in wavelet planes $\omega_l$ with $l = 2, \ldots, n_\omega$.

In the third and following columns in Table 4, we show the results obtained by WAVCLUS, K-MEANS, HIER and ROBUST methods, respectively.

Table 1
Correlation between consecutive approximation data sets $p_l$ for one-dimensional data set

| $p_5/p_4$ | $p_4/p_3$ | $p_3/p_2$ | $p_2/p_1$ | $p_1/p_0$ |
|---|---|---|---|---|
| 0.967 | 0.974 | 0.996 | 0.999 | 0.999 |

Original data set is $p_0$.

Table 2
Correlation between consecutive approximation data sets $p_l$ for two-dimensional data set

| $p_5/p_4$ | $p_4/p_3$ | $p_3/p_2$ | $p_2/p_1$ | $p_1/p_0$ |
|---|---|---|---|---|
| 0.931 | 0.938 | 0.988 | 0.998 | 0.999 |

Table 3
Correlation between consecutive approximation data sets $p_l$ for three-dimensional data set

| $p_5/p_4$ | $p_4/p_3$ | $p_3/p_2$ | $p_2/p_1$ | $p_1/p_0$ |
|---|---|---|---|---|
| 0.896 | 0.900 | 0.977 | 0.993 | 0.996 |

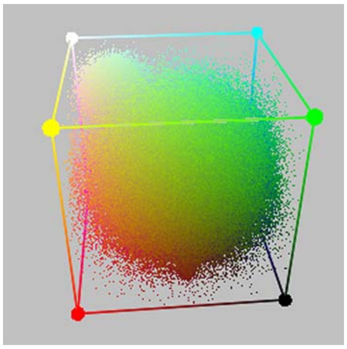

Fig. 3. Distribution of synthetic tridimensional data set in a tridimensional RGB space.

We proceeded similarly for a two and three-dimensional sets, showing the results in Tables 7 and 8, respectively. For the two and three-dimensional examples, we obtain $l_0 = 2$ and $l_0 = 1$ (Tables 2 and 3), respectively.

In Fig. 3 we show the distribution of the three-dimensional data set in its corresponding tridimensional space. In this figure, every dimension is displayed as a color axis, which allows to display the tridimensional data set in a RGB space. The axis values range from 0 to 32. In this figure we see the main cluster filling almost all the RGB space. One of the secondary clusters are located near the white vertex. We can also see the smaller cluster near the axis that goes from the black to the red vertex. These two secondary clusters are influenced by the presence of the main cluster, that is, they share some of the data points.

### 7.4. Discussion

In the second column of Table 4, we observe that cluster positions detected by the WAVCLUS method are exactly the original ones, but we have to be careful about this result. These clusters have exactly the same position as the true ones because the WAVCLUS method finds local maximums in a data set (the $d$-dimensional histogram) that is regularly sampled at values equal to the size of the bin used to obtain the histogram. Therefore, if the true cluster were located at a rational value (i.e., $x = 15.3$) we had not detected it exactly in this position, but had found it on the closer histogram bin value (i.e., $x = 15$). For a more accurate estimation of the cluster centroid, we could adjust a Gaussian profile or calculate a mass center using the points surrounding the local maximum. In this example, WAVCLUS uniquely detects these three clusters, showing that no other spurious or artifact clusters are present. The standard deviation of these clusters is relatively well estimated by the WAVCLUS method, even taking into account that it is an unsupervised method and that the two secondary clusters are embedded into the highly populated one, which hinders the estimation of the exact properties of the two small clusters.

Even if we assume that only three clusters are present in our data (which is seldom possible before an unsupervised classification), the K-MEANS algorithm can be forced to work with this number of clusters, placing it in a privileged position in front of the other methods. Even in this favorable initial situation, K-MEANS fails to find the clusters and their correct positions, and it distributes them almost uniformly along the numeric values. This can be seen on the standard deviation values calculated by K-MEANS for these clusters, being them almost the same. On the limit where the number of clusters is very high, the K-MEANS algorithm would distribute them uniformly through the $d$-dimensional space, finally obtaining a simple histogram regular interval decomposition devoid of useful information.

The percentile value of correctly classified pixels and the $\kappa$ index are also shown in Table 4 for each clustering method. These values show that classification accuracy of WAVCLUS method is better than K-MEANS.

The HIER and ROBUST methods have some problems for the one-dimensional data set. The ROBUST method only detects one cluster, which is a not a meaningful result. The HIER method overestimates the number of clusters, which in this case determines 15 clusters (these clusters are not shown to improve table readability and to save paper space). The $\kappa$ index shows that the classification

Table 4
Position and standard deviation of three clusters created in a 1,048,576 points unidimensional data set with dynamic range between 0 and 32

| | | True cluster | WAVCLUS | K-MEANS | HIER | ROBUST |
|---|---|---|---|---|---|---|
| Cluster statistics | A | $x = 15$, $\sigma = 5.0$, $p = 90\%$ | $x = 15$, $\sigma = 5.5$ | $x = 16.0$, $\sigma = 2.0$ | * | – |
| | B | $x = 25$, $\sigma = 2.0$, $p = 9\%$ | $x = 25$, $\sigma = 1.5$ | $x = 23.4$, $\sigma = 2.5$ | * | – |
| | C | $x = 5$, $\sigma = 1.0$, $p = 1\%$ | $x = 5$, $\sigma = 0.6$ | $x = 9.4$, $\sigma = 2.5$ | * | – |
| Accuracy | | | 82% | 52% | 8% | – |
| $\kappa$ Index | | | 0.41 | 0.19 | 0.02 | – |

In the corresponding clustering algorithm column, the percentile number of points belonging to a concrete cluster is shown. The accuracy index row show the number of pixels correctly classified. The last row is the value of $\kappa$ index for every clustering method. Character * means no data is shown (see text). Character – means no meaningful data is obtained.

accuracy is extremely poor. Confusion matrix is not shown for this method because it contains no useful information. One should take into account that the ROBUST method uses robust statistics for clustering determination. It implies that some small subclusters can be taken as outliers. On the other hand, the L-method used in the HIER algorithm tends to overestimate the number of clusters, which is an intrinsic property of the algorithm.

Table 5
Confusion matrix for the WAVCLUS one-dimensional dataset classification

| | | True cluster | | | |
|---|---|---|---|---|---|
| | | A | B | C | |
| Classified cluster | A | 756,575 | 7078 | 0 | 763,653 |
| | B | 78,840 | 87,294 | 0 | 166,134 |
| | C | 108,304 | 0 | 10,485 | 118,789 |
| | | 943,719 | 94,372 | 10,485 | 1,048,576 |

Table 6
Confusion matrix for the K-MEANS one-dimensional dataset classification

| | | True cluster | | | |
|---|---|---|---|---|---|
| | | A | B | C | |
| Classified cluster | A | 441,736 | 168 | 0 | 441,904 |
| | B | 188,407 | 94,204 | 0 | 282,611 |
| | C | 313,576 | 0 | 10,485 | 324,061 |
| | | 943,719 | 94,372 | 10,485 | 1,048,576 |

The confusion matrix of the WAVCLUS and K-MEANS classification algorithms, from which we have obtained accuracy percentages and the $\kappa$ index, is shown in Tables 5 and 6. In Table 6, where we show the confusion matrix for the K-MEANS classification, there is great confusion in the assignation of pixels to class A (first column), i.e., many pixels assigned to classes B and C truly belong to class A. The same confusion is shown by the WAVCLUS method, but this is due to the high standard deviation of class A. Clusters B and C are on the 'Gaussian wings' of cluster A, and some points belonging to A have the same values as the ones belonging to B and C.

Moreover, WAVCLUS method assigns around $7.6 \times 10^5$ points to cluster A, but K-MEANS assigns $4.4 \times 10^5$, being the true number around $9.4 \times 10^5$. This shows that WAVCLUS assigns a closer number of pixels to class A than K-MEANS, that is, K-MEANS shows a deficiency in the number of assigned pixels. The same behaviour is shown for classes B and C. From these observations, we can conclude the less relevant a cluster is, the more difficult to detect by the K-MEANS method.

In the two-dimensional data set, WAVCLUS method is again better than K-MEANS. The HIER method overestimates the number of clusters, similarly as with one-dimensional data set, and determines 16 clusters. Its $\kappa$ index is again close to zero. However, ROBUST method shows a classification accuracy a bit higher than the WAVCLUS method. As shown in Table 8, this method only detects two the three clusters, which are the most populated ones. But, in contrast to the WAVCLUS method, ROBUST method performs a better estimation of the cluster statistics. Standard deviations of clusters detected by this

Table 7
Position and standard deviation of three clusters created in a 1,048,576 points two-dimensional data set, with dynamic range between 0 and 32

| | True cluster | WAVCLUS | K-MEANS | HIER | ROBUST |
|---|---|---|---|---|---|
| Cluster statistics | $x = (15,15)$, $\sigma = (5.0,5.0)$, $p = 90\%$ | $x = (15,15)$, $\sigma = (6.7,6.8)$ | $x = (16.8,10.7)$, $\sigma = (3.7,3.4)$ | $*$ | $x = (15.2,15.0)$, $\sigma = (5.0,5.1)$ |
| | $x = (25,25)$, $\sigma = (2.0,2.0)$, $p = 9\%$ | $x = (25,25)$, $\sigma = (3.8,3.8)$ | $x = (22.0,21.8)$, $\sigma = (3.7,3.7)$ | $*$ | $x = (25.2,25.2)$, $\sigma = (2.0,2.1)$ |
| | $x = (15,5)$, $\sigma = (0.5,1.0)$, $p = 1\%$ | $x = (15,5)$, $\sigma = (2.2,2.4)$ | $x = (11.1,17.1)$, $\sigma = (3.4,3.7)$ | $*$ | – |
| Accuracy | | 85% | 47% | 9.2% | 96.8% |
| $\kappa$ Index | | 0.51 | 0.17 | −0.01 | 0.83 |

Character $*$ means no data is shown (see text). Character – means no meaningful data is obtained.

Table 8
Position, standard deviation and percentile number of pixels for three clusters created in a 1,048,576 points three-dimensional data set, with dynamic range between 0 and 32

| | True cluster | WAVCLUS | K-MEANS | HIER | ROBUST |
|---|---|---|---|---|---|
| Cluster | $x = (15,15,15)$, | $x = (15,15,15)$, | $x = (17.5,12.0,12.9)$ | – | $x = (15.0,14.7,14.8)$, |
| statistics | $\sigma = (5,5,5)$, $p = 90\%$ | $\sigma = (7.3,7.3,7.4)$ | $\sigma = (3.7,4.2,4.6)$ | | $\sigma = (5.0,4.9,5.1)$ |
| | $x = (25,25,25)$, | $x = (25,25,25)$, | $x = (22.7,22.5,22.5)$, | – | $x = (25.0,24.7,24.6)$, |
| | $\sigma = (2,2,2)$, $p = 9\%$ | $\sigma = (3.6,3.6,3.6)$ | $\sigma = (3.6,4.1,4.1)$ | | $\sigma = (2.1,2.3,2.2)$ |
| | $x = (15,5,5)$, | $x = (15,5,5)$, | $x = (11.5,16.6,15.9)$, | | |
| | $\sigma = (0.5,1,1)$, $p = 1\%$ | $\sigma = (2.3,2.4,2.4)$ | $\sigma = (3.5,4.2,4.6)$ | | |
| Accuracy | | 96% | 52% | 2.9% | 97.5% |
| $\kappa$ Index | | 0.82 | 0.21 | −0.01 | 0.86 |

Character $*$ means no data is shown (see text). Character – means no meaningful data is obtained.

method are better than the one estimated by the WAVCLUS, which produces a much better classification accuracy. Hence, the WAVCLUS method is better detecting the real clusters, but ROBUST method is better determining cluster statistics.

In the three-dimensional data set, this behaviour appears again, see Table 8. HIER method detects 26 clusters. ROBUST method detects only the two most populated clusters, but their statistics are much better estimated than by WAVCLUS method.

## 8. Real data

In order to test the accuracy of WAVCLUS in front of real data, a supervised classification of two real images, *ball1* and *ball2*, see Fig. 4(a) and (b) respectively, was performed.

### 8.1. Classification process

We performed a supervised classification using WAVCLUS, HIER and ROBUST algorithms for the initial unsupervised classification step usually present into supervised classification processes. We performed this supervised classification in the usual two-step process: (i) perform a unsupervised classification, obtaining a number $n_c$ of classes, and (ii) perform a final supervised classification step using the previous $n_c$ classes as input, obtaining finally $m$

Table 9
Correlation between consecutive approximation data sets $p_l$ for *ball1*

| $p_5/p_4$ | $p_4/p_3$ | $p_3/p_2$ | $p_2/p_1$ | $p_1/p_0$ |
|---|---|---|---|---|
| 0.726 | 0.728 | 0.741 | 0.738 | 0.764 |

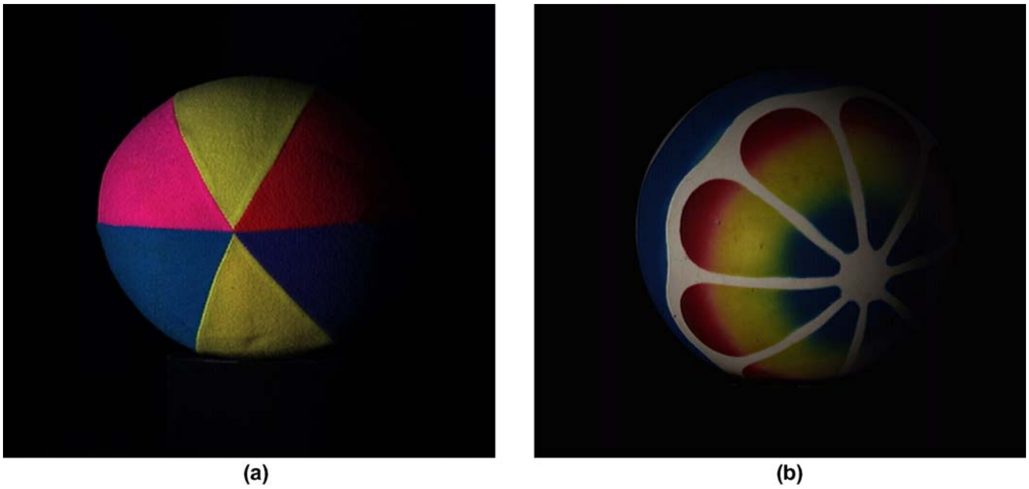

Fig. 4. Real color images of two balls used as input to several classification algorithms. (a) *ball1* and (b) *ball2*.

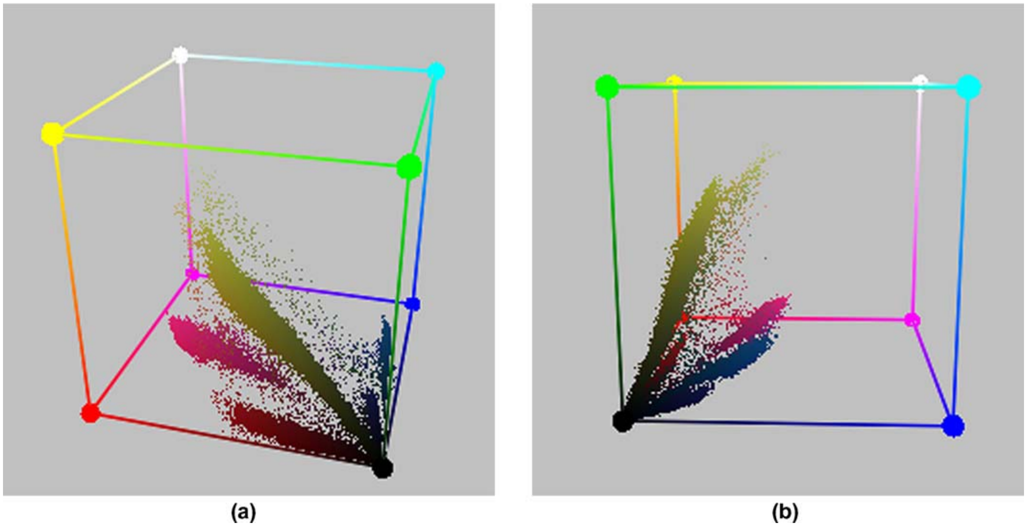

Fig. 5. Two different points of view in the RGB space of the *ball1* data set.

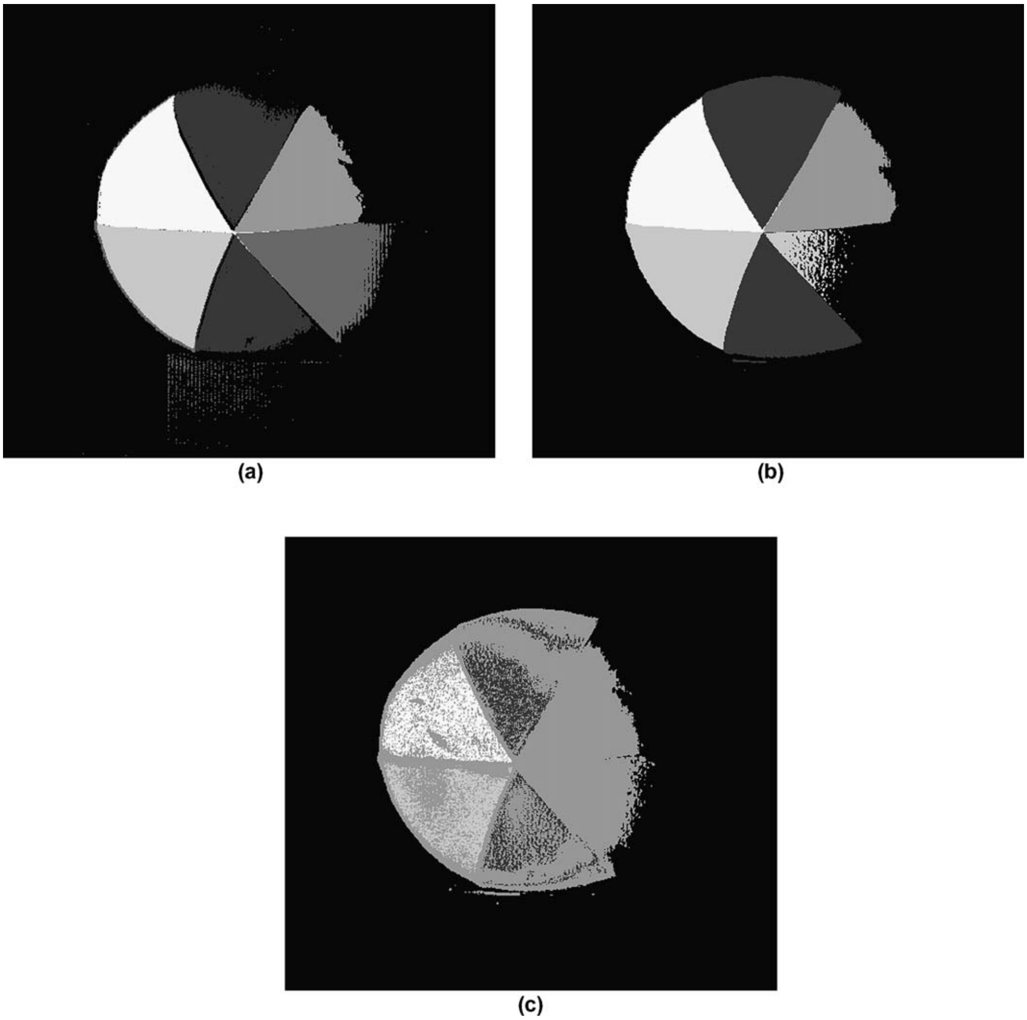

Fig. 6. Final supervised classification of *ball1* for the corresponding clustering methods. (a) WAVCLUS method, (b) HIER method and (c) ROBUST method.

groups as output. These $m$ groups were previously defined by a human operator as training regions for the supervised classification. Six and seven groups were defined for the *ball1* and *ball2* images, respectively.

### 8.2. Results for ball1

In Fig. 5(a) and (b) we show two different points of view of the tridimensional representation of the data set in the RGB space. In these images we can see the several clusters present in our data set: a long and narrow yellow-green[1] cluster, a pink cluster, a red cluster and two blue clusters. These clusters correspond to the different colors of the patches present in the ball. These visible clusters are the ones that the clustering methods should detect.

For this case we take $s_b = 2$, which leads to a tridimensional histogram with $128^3$ bins. From the correlation values shown in Table 9, we obtain $l_0 = 2$ because in this case there is a negative increment of the correlation between $p_2/p_3$ and $p_1/p_2$. We show in Fig. 6(a)–(c) the WAVCLUS, HIER and ROBUST supervised classification results, respectively.

The WAVCLUS, HIER and ROBUST methods automatically obtained 7, 8 and 9 classes, respectively, during the unsupervised classification process for this image. For the final supervised classification images (see Fig. 6), we show in Table 11 the $\kappa$ index and the percentage of correctly classified pixels.

In Table 10 we show the clusters detected by the WAVCLUS method during the unsupervised step. Since the feature space is the RGB space, the position and square root of variances of the clusters are described in this space coordinates. Comparing these values with the data distribution in Fig. 5, we can see that every cluster detected by the WAVCLUS method corresponds to one of the clusters present in this figure. The axis in Fig. 5 range from 0 to 255. The first cluster corresponds to the black background

[1] For interpretation of references in color, the reader is referred to the web version of this article.

Table 10
Cluster statistics, position and square root of variance, obtained by WAVCLUS for *ball1* image

| R | G | B | $\sigma_R$ | $\sigma_G$ | $\sigma_B$ |
|---|---|---|---|---|---|
| 2 | 2 | 2 | 2.05 | 2.00 | 1.98 |
| 10 | 48 | 92 | 9.10 | 8.90 | 9.32 |
| 12 | 12 | 46 | 9.08 | 9.12 | 9.72 |
| 40 | 42 | 18 | 9.76 | 9.78 | 8.38 |
| 74 | 74 | 28 | 9.70 | 9.48 | 8.24 |
| 108 | 106 | 36 | 9.62 | 9.54 | 8.14 |
| 201 | 24 | 98 | 9.48 | 8.34 | 8.90 |

Table 11
$\kappa$ index and classification accuracy (percentage of correctly classified pixels) for the supervised classification of *ball1* image

| *ball1* | WAVCLUS | HIER | ROBUST |
|---|---|---|---|
| $\kappa$ | 0.87 | 0.81 | 0.59 |
| Accuracy (%) | 94.7 | 92.8 | 84.4 |

which is not visible in this figure because is extremely concentrated at the origin, the second one to the more blue cluster, the third to the small darker blue cluster which is a bit difficult to see in this figure, the fourth to a half of the big yellow-green cluster, the fifth to the other half of the same cluster, the sixth to the red cluster, and the seventh to the pink one. Hence, we can see that the WAVCLUS method is able to detect in a very acceptable fashion the clusters present in our data.

In Table 11 we can see that WAVCLUS obtains the highest $\kappa$ value, being HIER the following and ROBUST the worst. Nevertheless, even being the WAVCLUS accuracy higher than the HIER method, we consider that the difference between the WAVLCUS and the HIER $\kappa$ index is not very significant, and that we should consider that both methods obtain similar results. Tables 12–14 show the confusion matrices of the corresponding methods when comparing the supervised classification image with a truth data set defined by a human operator. Looking at the final supervised classification images in Fig. 6 we can see that WAVCLUS and HIER obtains a good result, consistently classifying every color patch of the ball in a separate class. The HIER method does not obtain a good result on the dark blue patch, which is confused with the dark

Table 12
Confusion matrix for the WAVCLUS supervised classification of the *ball1* image

| *ball1*/WAVCLUS | Background | Yellow | Dark blue | Red | Blue | Pink | |
|---|---|---|---|---|---|---|---|
| Background | 177,386 | 0 | 0 | 0 | 0 | 0 | 177,386 |
| Yellow | 4421 | 13,735 | 0 | 685 | 0 | 0 | 18,841 |
| Dark blue | 1975 | 0 | 7959 | 203 | 0 | 0 | 10,137 |
| Red | 1808 | 1494 | 0 | 8376 | 0 | 0 | 11,678 |
| Blue | 666 | 0 | 0 | 0 | 8601 | 0 | 9267 |
| Pink | 1295 | 0 | 0 | 3 | 0 | 8669 | 9967 |
| | 187,551 | 15,229 | 7959 | 9267 | 8601 | 8669 | 237,276 |

Table 13
Confusion matrix for the HIER supervised classification of the *ball1* image

| *ball1*/HIER | Background | Yellow | Dark blue | Red | Blue | Pink | |
|---|---|---|---|---|---|---|---|
| Background | 178,602 | 0 | 6651 | 211 | 0 | 0 | 185,464 |
| Yellow | 4912 | 15,229 | 0 | 0 | 0 | 0 | 20,141 |
| Dark blue | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Red | 1331 | 0 | 0 | 9056 | 0 | 0 | 10,387 |
| Blue | 1299 | 0 | 1308 | 0 | 8601 | 0 | 11,208 |
| Pink | 1407 | 0 | 0 | 0 | 0 | 8669 | 10,076 |
| | 187,551 | 15,229 | 7959 | 9267 | 8601 | 8669 | 237,276 |

Table 14
Confusion matrix for the ROBUST supervised classification of the *ball1* image

| *ball1*/ROBUST | Background | Yellow | Dark blue | Red | Blue | Pink | |
|---|---|---|---|---|---|---|---|
| Background | 177,790 | 0 | 1830 | 1632 | 0 | 0 | 181,252 |
| Yellow | 1274 | 4811 | 0 | 0 | 0 | 0 | 6085 |
| Dark blue | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Red | 7937 | 10,418 | 6129 | 7635 | 4318 | 2971 | 39,408 |
| Blue | 113 | 0 | 0 | 0 | 4283 | 0 | 4396 |
| Pink | 437 | 0 | 0 | 0 | 0 | 5698 | 6135 |
| | 187,551 | 15,229 | 7959 | 9267 | 8601 | 8669 | 237,276 |

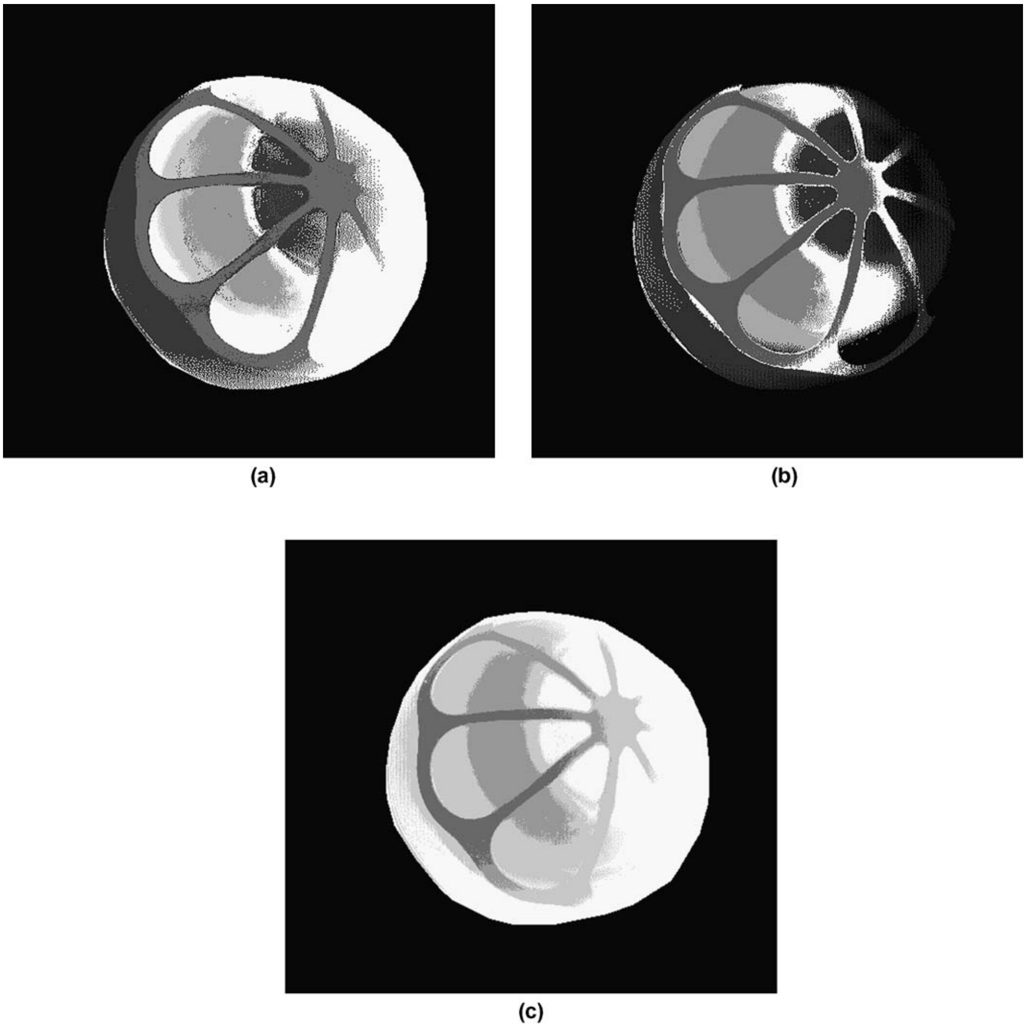

Fig. 7. Final supervised classification of *ball2* for the corresponding clustering methods. (a) WAVCLUS method, (b) HIER method and (c) ROBUST method.

background and the blue patch. It can be seen on the confusion matrix: the WAVCLUS wrongly classifies 28 dark blue pixels as background pixels (Table 12); the HIER method wrongly classifies the dark blue pixels as background (6651 pixels) and blue (1308 pixels) and does not classify any pixel as dark blue (Table 13). The rest of groups are approximately correctly classified by WAVCLUS and HIER. WAVCLUS confusion matrix is nearly diagonal, while HIER matrix shows more confusion. Both methods show some confusion on the background group.

The classification obtained by the ROBUST method shows great confusion. Red and dark blue groups are confused, as well as the result is highly affected by small shadows in the rough surface. Confusion matrix is not nearly diagonal because of the read and blue groups.

### 8.3. Results for ball2

For this case we take $s_b = 2$. For this case we obtain $l_0 = 2$. We show in Fig. 7(a)–(c) the WAVCLUS, HIER and ROBUST supervised classification results, respectively.

The WAVCLUS, HIER and ROBUST methods automatically obtained 8, 9 and 7 classes, respectively, during the unsupervised classification process for this image. For the final supervised classification results, we show in Table 15 the $\kappa$ index and the percentage of correctly classified pixels.

Tables 16–18 show the confusion matrices of the corresponding methods.

In this example, WAVCLUS accuracy is a bit lower than HIER but we consider, similarly to *ball1* example,

Table 15
$\kappa$ index and classification accuracy (percentage of correctly classified pixels) for the supervised classification of *ball2* image

| *ball2* | WAVCLUS | HIER | ROBUST |
|---|---|---|---|
| $\kappa$ | 0.68 | 0.73 | 0.59 |
| Accuracy (%) | 82.0 | 85.7 | 76.7 |

Table 16
Confusion matrix for the WAVCLUS supervised classification of the *ball2* image

| *ball2*/WAVCLUS | C1 | C2 | C3 | C4 | C5 | C6 | C7 | |
|---|---|---|---|---|---|---|---|---|
| C1 | 151,289 | 357 | 53 | 0 | 0 | 82 | 0 | 151,781 |
| C2 | 22 | 9522 | 1144 | 22 | 71 | 0 | 146 | 10,927 |
| C3 | 62 | 504 | 15331 | 73 | 416 | 0 | 519 | 16,905 |
| C4 | 0 | 0 | 231 | 10011 | 6 102 | 0 | 313 | 10,657 |
| C5 | 0 | 0 | 98 | 903 | 3388 | 0 | 1 | 4390 |
| C6 | 1776 | 11,345 | 5505 | 7666 | 9134 | 5092 | 2098 | 42,616 |
| C7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 153,149 | 21,728 | 22,362 | 18,675 | 13,111 | 5174 | 3077 | 237,276 |

Table 17
Confusion matrix for the HIER supervised classification of the *ball2* image

| *ball2*/HIER | C1 | C2 | C3 | C4 | C5 | C6 | C7 | |
|---|---|---|---|---|---|---|---|---|
| C1 | 152,884 | 3528 | 1264 | 397 | 3159 | 3530 | 51 | 164,813 |
| C2 | 167 | 15,734 | 2492 | 625 | 90 | 1631 | 477 | 21,216 |
| C3 | 28 | 193 | 13,441 | 22 | 292 | 0 | 23 | 13,999 |
| C4 | 1 | 1 | 915 | 12,363 | 2212 | 0 | 502 | 15,994 |
| C5 | 0 | 0 | 39 | 27 | 6970 | 0 | 0 | 7036 |
| C6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C7 | 69 | 2272 | 4211 | 5241 | 388 | 13 | 2024 | 14,218 |
| | 153,149 | 21,728 | 22,362 | 18,675 | 13,111 | 5174 | 3077 | 237,276 |

Table 18
Confusion matrix for the ROBUST supervised classification of the *ball2* image

| *ball2*/ROBUST | C1 | C2 | C3 | C4 | C5 | C6 | C7 | |
|---|---|---|---|---|---|---|---|---|
| C1 | 151,289 | 357 | 53 | 0 | 0 | 82 | 0 | 151,781 |
| C2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C3 | 0 | 32 | 6513 | 4 | 65 | 0 | 7 | 6621 |
| C4 | 0 | 27 | 3657 | 9666 | 204 | 0 | 117 | 13,671 |
| C5 | 73 | 3861 | 7335 | 4608 | 9411 | 0 | 1539 | 26,827 |
| C6 | 1787 | 17451 | 4804 | 4397 | 3431 | 5092 | 1414 | 38,376 |
| C7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 153,149 | 21,728 | 22,362 | 18,675 | 13,111 | 5174 | 3077 | 237,276 |

that this difference is not very relevant. Their $\kappa$ index are relatively similar, and they are greater than the one for ROBUST method. From confusion matrices, we can see that there is great confusion in C6 and C7 groups in all three methods, and the HIER method also shows confusion on the C1 group.

## 9. Conclusions

As can be seen from the synthetic examples, WAVCLUS method usually detects the correct number of clusters and obtains an acceptable statistics of these clusters. In comparison, the HIER method detects the wrong number of clusters. In fact, it performs an overestimation, and its classification accuracy is low. On the other side, the ROBUST method detects some of the clusters, but cannot detect the less populated ones, i.e., it underestimates the number of clusters; in contrast it better estimates the cluster statistics, which produces a better final classification accuracy. It suggest that WAVCLUS trades accuracy in the statistics of the clusters for the estimation of the number of clusters. As a result, due to its robustness, WAVCLUS is always applicable to a vast variety of scenarios.

As shown with real data, which is used to perform a supervised classification, both WAVCLUS and HIER method show similar final classification accuracies, except for the ROBUST method which shows a worse classification accuracy.

Since WAVCLUS automatically determines the number of clusters, it is more appropriate when the user has no a priori information on the number of clusters present in the images. It usually occurs when great amounts of data need a first blind classification or fast analysis.

However, the WAVCLUS method requires to take into account several advices. Since it works with the $d$-dimensional histogram, the user has to be aware of any previous histogram or data manipulation. Usual histogram stretching or expansion for image contrast improvement and visualization are misleading and WAVCLUS data classification of such images may supply meaningless clusters. If we regard the histogram as a mathematical function, the worst situation is that after modifying the histogram some null values are obtained, i.e., there are no data points assigned to a concrete $d$-dimensional value. In this case, the WAVCLUS method detects the sudden change in the histogram and considers it as an evidence of cluster. Another misleading situation is encountered when the histogram is a not-smooth function and it presents sudden changes (the former situation would be a particular case), not necessarily involving null values. This situation is usually linked to histogram stretching. These sudden changes, usually manifested as peaks or valleys, are interpreted as clusters by the WAVCLUS method. The performance of classification methods not based on histogram processing may be also affected by these manipulations but not to the same extent as the WAVCLUS. This is a drawback of the WAVCLUS method, but with careful manipulation of the original data, the method proves to be robust, reliable and accurate enough when compared to other well established methods.

Therefore, the general wavelet based model we presented in Section 4 seems to be able to automatically detect the number of clusters present in our multidimensional data and estimate their statistical properties, working only with the data to analyze and the minimum correlation value supplied by human operator.

When using the à *trous* algorithm to perform the wavelet transform, several drawbacks appeared on this particular implementation (see Section 10 for computer issues), which leads us to work only with low-dimensional multispectral data sets. Other wavelet transform algorithms can be used to reduce the number of data to process, but the described general wavelet based model is the general guide to perform a cluster analysis of the data.

## 10. Computer implementation

As discussed above, there are several drawbacks on the computer implementation of the WAVCLUS algorithm. The reason of these problems is the dimensionality of data. Given a multispectral ($d$-dimensional) data set, WAVCLUS has to compute the $d$-dimensional histogram.

### 10.1. Memory storage

Let the histogram contains 128 bins in every dimension, hence the $d$-dimensional histogram has $n = 128^d$ data points. For $d = 3$, $n \sim 2.0 \times 10^6$ data points, and storing them in floating point registers (4 bytes), requires 8 Mbytes. We have to multiply this value to store every wavelet plane and many temporary data arrays. In practice, this leads to some hundreds of Mbytes of memory. If $d = 4$ the final amount of data would really be unmanageable.

Therefore, it is necessary to reduce the amount of data to store and to process. Two main strategies can be used:

- Reduce dimensionality of data.
- Store only useful data.

Reduction of dimensionality can be approached using PCA analysis, only retaining the few most significant channels. Obviously, classification accuracy is reduced in a significant way, and as much dimensionality reduction is performed much lower will be the classification accuracy. Many useful multispectral images, including hyperspectral ones, are 'dimensionally' far beyond the computing possibilities of WAVCLUS algorithm as presented above, but another approach not based on dimensionality reduction can be used.

Another possibility to reduce the amount of data is to store only the useful histogram bins, in other words, the non-null histogram values. Most of the bins are not populated by any data point, and these points are not used in any computation, so it is not necessary to store this values in memory. On the other hand, the higher the dimensionality of the multispectral data set, the sparser the histogram. In a extreme situation the histogram may be so sparse that the histogram will not be useful, but this is the well known problem of sparsity and dimensionality. This approach has another limitation: we have to work only on the bins where there is data, which means it is not possible to 'interpolate' values among them (wavelet transform does this). We are forced to work all the time only on not null bins, limiting the possibility to interpolate values among them to increase data processing and localization of local maximums. WAVCLUS would find centroids only on those multispectral points present in the original data.

But in real data, multispectral centroids are very close to data points, so it is very likely that many original multispectral data points from our original multispectral image to classify are very close or equal to centroids. Therefore, to work only on not null bins does not has to be a problem. This assumption is too strong for a general pattern analysis method, but not so restrictive for real applications where data is clustered around a central point (the multispectral centroid). We have not used this strategy in this work, so this is a point for future work and computer implementations of WAVCLUS.

This last approach could be very useful also to reduce the extremely long CPU-time used by the general WAVCLUS algorithm.

As stated in previous section, the most feasible solution to the general problem, seems to be the using of other wavelet decomposition algorithms different to the à *trous* algorithm.

## Acknowledgements

Authors want to thank anonymous reviewers for their helpful comments and suggestions in order to improve the quality of both the work and the paper. Authors want to thank Fernando Vilariño for his helpful comments on clustering techniques. Xavier Otazu wants to thank Ramon y Cajal programme for its economical support. This research was performed using computing facilities at CEPBA

(European Center for Parallelism of Barcelona). This work has been partially supported by project TIN2004-02970 of Spanish MEC (Spanish Ministery of Science).

## References

Chui, C.K., 1992. An Introduction to Wavelets. Boston Ac. Press, Boston.

Daubechies, I., 1992. Ten Lectures on Wavelets. SIAM Press, Philadelphia.

Hartigan, J.A., 1975. Clustering Algorithms. Wiley.

Holschneider, M., Tchamitchian, P., 1990. Les ondelettes en 1989. In: Lemarié, P.G. (Ed.). Springer-Verlag, Paris.

Kaiser, G., 1994. A Friendly Guide to Wavelets. Birkhauser, Boston.

Kaufman, L., Rosseeuw, P.J., 1990. Finding Groups in Data: An Introduction to Cluster Analysis. Wiley.

Kohonen, T., 1988. Learning vector quantization. Neural Networks 1, 303.

Letts, P.A., 1978. Unsupervised classification in the aries image analysis system. Proc. 5th Can. Symp. on Remote Sensing, 61–71.

Mallat, S., 1998. A Wavelet Tour of Signal Processing, second ed. Academic Press, San Diego.

McLachlan, G.J., Basford, K.E., 1988. Mixture Models: Inference and Applications to Clustering. Marcel Dekker, New York.

Meyer, Y., 1993. Wavelets: Algorithms and Applications. SIAM Press, Philadelphia.

Yang, Miin-Shen, Wu, Kuo-Lung, 2004. A similarity-based robust clustering method. IEEE Trans. Pattern Recognition Machine Intell. 26 (4), 434–448.

Rué, F., Bijaoui, A., 1996. A multiscale vision model applied to astronomical images. Vistas Astron. 40, 495–502.

Salvador, S., Chan, P., 2004. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. Proc. 16th IEEE Internat. Conf. on Tools with AI, 576–584.

Smyth, P., 1996. Clustering using Monte-Carlo cross-validation. Proc. KDD, 126–133.

Starck, J.L., Murtagh, F., 1994. Image restoration with noise suppression using the wavelet transform. Astron. Astrophys. 288, 342–350.

Starck, J.L., Pantin, E., 1996. Multiscale maximum entropy image restoration. Vistas Astron. 40, 563–569.

Starck, J.L., Murtagh, F., Bijaoui, A., 1998. Image and data analysis: the multiscale approach. Cambridge University Press, Cambridge.

Sugiyama, M., Ogawa, H., 2001. Subspace information criterion for model selection. Neural Comput. 13 (8), 1863–1889.

Tibshirani, R., Walther, G., Hastie, T., 2003. Estimating the number of clusters in a data set via the GAP statistic. JRSSB.

Vetterli, M., Kovacevic, J., 1995. Wavelets and subband coding. Prentice Hall.