# Deliverable R3–B1–P
# Task B1 : Databases

Covering the period from: July, 1st, 1994 to: June, 30th, 1995

ESPRIT Basic Research Project Number 6891

ELENA

# Enhanced Learning for Evolutive Neural Architecture


Project Coordinator
C. Jutten (INPG)

Task leader
A. Guérin-Dugué (INPG)

Authors
C. Aviles-Cruz, A. Guérin-Dugué (INPG)
J.L. Voz (UCL)
D. Van Cappel (TSA)



June 30, 1995

**Abstract**

Task B1 of the Elena project is aimed to provide a set of databases to be used for tests and benchmarks of the neural and classification algorithms studied and developed in the project. This report describes all databases selected for this purpose. They are splitted into two parts: artificially generated databases, mainly used for preliminary tests, and real ones, used for objective benchmarks and comparisons of methods.

The choice of the database has been guided by various parameters, such as availability of published results concerning conventional classification algorithms, size of the database, number of attributes, number of classes, overlapping between classes and non-linearities of the borders,...Results of PCA and DFA preprocessing of the real databases are also included, together with several measures useful for the databases characterization (statistics, fractal dimension, dispersion,...).

All these databases and their preprocessing are available to the partners of the project on a file server located in Louvain-la-Neuve (Belgium). They will also be made available for anonymous ftp at the "UCI Repository Of Machine Learning Databases and Domain Theories" [20] together with this report and the report related to the comparative benchmarking studies (R3-B4-P).

# Contents

# Chapter 1

# Introduction

## 1.1 Choice of the databases

Testing existing and new developed algorithms requires to have databases at disposal, on which tests and benchmarks of the methods can be realized. Some database sets where recently proposed as standard benchmark sets for classifier learning. Let us cite for example [26] and also the Proben1 set [22] which is a collection of problems for neural network learning in the realm of pattern classification and function approximation plus a set of rules and conventions for carrying out benchmark tests with these or similar problems. Since these sets where not available at the beginning of the project, the ELENA partners set up their own database set, the choice having been guided by various parameters, such as availability of published results concerning conventional classification algorithms, size of the database, number of attributes, number of classes, overlapping between classes and non-linearities of the borders,...

The internal meeting in Lausanne (September 1994) was the opportunity to fix definitely the set of databases used by the different partners in the project; the use of identical databases by all partners is obviously intended to allow objective comparisons of methods. In this last report related to the ELENA databases, we describe in details all databases that the partners decided to keep for classifers benchmarking studies.

The databases are splitted into two parts: the *artificial* ones, being generated in order to obtain some defined characteristics, and for which the theoretical Bayes error can be computed, and the *real* ones, collected in existing applications or coming from the "UCI Repository Of Machine Learning Databases and Domain Theories" [20].

- The *artificial* databases (*Gaussian, Clouds* and *Concentric*) were generated according to the following requirements:
  - heavy intersection of the class distributions,
  - high degree of nonlinearity of the class boundaries,
  - various dimensions of the vectors,
  - already published results on these databases.

  They are restricted to two-class problems, since we believe it yield answers to the most essential questions [17]. The artificial databases are mainly used for rapid test purposes on newly developed algorithms.

- The *real* databases (*Satimage, Texture, Iris* and *Phoneme*) were selected according to the following requirements:
  - classical databases in the field of classification (*Iris*),
  - already published results on these databases (*Phoneme*, from the ROARS ESPRIT project [1] and *Satimage* from the STATLOG ESPRIT project [21] [19]),
  - various dimensions of the vectors,
  - sufficient number of vectors (to avoid the "empty space phenomenon").
  - the *Texture* database, generated at INPG for the Elena project is interesting for its high number of classes (11).

The R3-B4-P ELENA report is related to the benchmarking studies of various classifiers either well-known by the Neural Network community (MLP, RCE, LVQ, k_NN, GQC) either developped in the framework of the project (IRVQ, PLS). It will also be made available on the public domain. Its main objectives are the following.

- to provide an overall comprehensive view of the general problem of comparative benchmarking studies and to propose a useful common test basis for existing and further classification methods,
- to obtain objective comparisons of the different chosen classifiers on the set of databases described in this report (each classifier being used with its optimal configuration for each particular database),
- to study the possible links between the data structures of the databases viewed by some parameters, and the behavior of the studied classifiers (mainly the evolution of their the optimal configuration parameters).
- to study the links between the preprocessing methods and the classification algorithms from the performances and hardware constraints point of view (especially the computation times and memory requirements).

## 1.2 Predefined internal data format

During the second meeting between the different partners involved in the project, an internal format for all databases used in the project was defined: it is referenced in the different deliverables of this Progress Report as the *Elena database format.*

Here is its description:

- All files containing the databases are stored as ASCII files for their easy edition and checking.
- In a file, each of the $n$ lines is reserved for each vectorial sample (instance) and each line consists of $d$ floating-point numbers (the attributes) followed by the class label (which must be an integer).

**Example :**

Here is an example of a small database in the Elena format containing $n = 6$ instances with $d = 9$ attributes plus the class value at the last column:

```
1.51768 12.65 3.56 1.30 73.08 0.61 8.69 0.00 0.14 1
1.51747 12.84 3.50 1.14 73.27 0.56 8.55 0.00 0.00 0
1.51775 12.85 3.48 1.23 72.97 0.61 8.56 0.09 0.22 1
1.51753 12.57 3.47 1.38 73.39 0.60 8.55 0.00 0.06 1
1.51783 12.69 3.54 1.34 72.95 0.57 8.75 0.00 0.00 3
1.51567 13.29 3.45 1.21 72.74 0.56 8.57 0.00 0.00 1
```

A practical interface between databases in the Elena format and the software simulators developed in the PACKLIB unified graphic environment is available in the software module GENE (see report of task B3).

## 1.3 Files organization on the server

This section provides useful information on the method to get the currently available databases of the project as well as their preprocessing. It serves as the S2-B1-R Report (files containing the databases).

A ftp server is available to the ELENA partners to allow an optimum cooperation between all the partners of the Elena project. The password to access this server was provided to the partners during the first meeting. The ELENA databases will also be made available for anonymous ftp on the "UCI Repository Of Machine Learning Databases and Domain Theories" [20] together with the R3-B4-P report and this one.

The databases are grouped on the Elena server in the so-called *databases* directory (and in the *elena* directory of the anonymous UCI server). This directory contains a *README* file and two subdirectories: *ARTIFICIAL* and *REAL*. The *README* file is a text file providing information on the contents of the *databases* (or *elena*) directory.

### 1.3.1 Contents of the ARTIFICIAL directory

The databases of this directory contain only the artificial classification problem sets selected during the January 94 meeting in Chamrousse (the official databases of the Elena project).

These artificial databases are only two-class problems, since it yields answers to the most essential questions. For each problem, the confusion matrix corresponding to the theoretical Bayes boundary is provided with the confusion matrix obtained by a k_NN classifier (a simple majority voting between the k nearest prototypes of the point to classify, k chosen to reach the minimum of the total Leave-One-Out error).

There is one subdirectory for each database. In this subdirectory, there are :

- a text file providing detailed information about the related database ("database-name.txt"),
- the compressed database in the Elena format ("databasename.dat.Z"); the different patterns of each database are presented in a random order,

- for bidimensional databases, a postscript file representing the 2-D datasets (those files are in eps format generated with the PACKLIB Neural Network graphical simulation tool developped in the framework of the Elena project [2], you can print them on any ps printer or simply have a look on your SUN workstation with any postscript previewer).

For each subdirectory, the directoryname is the same as the name chosen for the concerned database: *concentric, clouds* and *gaussian.*

### 1.3.2 Contents of the REAL directory

The databases of this directory contain only the real classification problem sets selected for the ELENA benchmarking studies. There is one subdirectory for each database. In this subdirectory, there are:

- a text file giving detailed information about the related database ("databasename.txt"),
- the compressed original database in the Elena format ("databasename.dat.Z"); the different patterns of each database being presented in a random order.
- By the way of a normalization process, each original feature will have the same importance in a subsequent classification process. A typical method is first to center each feature separately an than to reduce it to a unit variance; this process is recommended when the original features have very different scales. It has been applied on all the *real* Elena databases in order to build the "CR" databases contained in the "databasename_CR.dat.Z" files.
- The Principal Components Analysis is a very classical method in pattern recognition [9]. PCA reduces the sample dimension in a linear way for the best representation in lower dimensions keeping the maximum of inertia. The best axe for the representation is however not necessary the best axe for the discrimination. After PCA, features are selected according to the percentage of initial inertia which is covered by the different axes and the number of features is determined according to the percentage of initial inertia to keep for the classification process. This selection method has been applied on the *satimage_CR* and *texture_CR* databases. When quasi-linear correlations exists between some initial features, these redundant dimensions are removed by PCA and this preprocessing is then recommended. In this case, before a PCA, the determinant of the data covariance matrix is near zero; this database is thus badly conditioned for all process which use this information (the quadratic classifier for example). The following files are available for the *real* databases:
  - "databasename_PCA.dat.Z", the projection of the "CR" database on its principal components (sorted in a decreasing order of the related inertia percentage),
  - "databasename_corr_circle.ps", a graphical representation of the correlation between the initial attributes and the two first principal components,

  - "databasename_proj_PCA.ps", a graphical representation of the projection of the initial database on the two first principal components,
  - "databasename.txt", a file with the eigenvalues and associated inertia percentages

- The Discriminant Factorial Analysis can be applied to a learning database where each learning sample belongs to a particular class [9]. The number of discriminant features selected by DFA is fixed in function of the number of classes ($c$) and of the number of input dimensions ($d$); this number is equal to the minimum between $d$ and $c-1$. In the usual case where $d$ is greater than $c$, the output dimension is fixed equal to the number of classes minus one and the discriminant axes are selected in order to maximize the between-variance and to minimize the within-variance of the classes. The discrimination power (ratio of the projected between-variance over the projected within-variance) is not the same for each discriminant axis: this ratio decreases for each axis. So for a problem with many classes, this preprocessing will not be always efficient as the last output features will not be so discriminant. This analysis uses the information of the inverse of the global covariance matrix, so the covariance matrix must be well conditioned (for example, a preliminary PCA must be applied to remove the linearly correlated dimensions). The DFA preprocessing method has been applied on the 18 first principal components of the *satimage_PCA* and *texture_PCA* databases [1] in order to build the *satimage_DFA.dat.Z* and *texture_DFA.dat.Z* database files, having respectively 5 and 10 dimensions [2].

For each subdirectory, the directoryname is the same as the name chosen for the contained database: *satimage, phoneme, iris* and *texture*.

## 1.4 A classical ftp session

Here is a reminder of the ftp method the partners have to use to get a database on the ELENA ftp server (or on the UCI anonymous ftp server):

ftp satie.dice.ucl.ac.be (*or* ftp ics.uci.edu)
*Name?* elena (*or* anonymous)
*Password?* N..... (*or your e-mail address*)
binary (don't forget this command)
cd databases (*or* cd pub/machine-learning-databases/elena)
get README (then read this file to know what you want to get)
cd *directoryname*
cd *directoryname*
get *databasename.txt*
get *databasename.data.Z*
...
quit

[1] Thus by keeping only the 18 first attributes of these databases before to apply the DFA preprocessing

[2] The *satimage* database having 6 classes and *texture* 11

After your ftp session, just type (on your SUN workstation):

uncompress *databasename.data.Z*

to get the uncompressed datafile.

## 1.5 Organization of the report

The two next chapters of this report respectively present the *artificial* and *real* databases. Each database corresponds to a section in the concerned chapter. Each artificial database is described by the following information: its aim (the type of tests which can be carried out), its description, the theoretical confusion matrix obtained with a Bayesian classifier, and the Leave-One-out confusion matrix obtained with a classical k_NN classifier (as a reference for tests). The k_NN classifier consist in a simple majority voting between the k nearest prototypes of the point to classify, k chosen to reach the minimum of the total Leave-One-Out error.

Each real database is described by the following information:

1. the source (where it does come from),
2. relevant information,
3. statistics (mean, variance, correlations),
4. the attribute precision,
5. relative number of samples (see section 1.5.1),
6. estimate of the fractal dimension (see section 1.5.2),
7. inertia and samples dispersion (see section 1.5.3),
8. the confusion matrix with confidence intervals obtained with a k_NN classifier (see section 1.5.4),
9. the results of a Principal Component Analysis.

### 1.5.1 Relative number of available data

In report R1-A-P, the minimum number of samples necessary for the estimation of the *probability density function* of a Gaussian underlying density by a probability density kernel estimator with less than 10% of error have been evaluated by :

$$log_{10}N \approx 0.6(d - \frac{1}{4})$$

For classification methods based on kernel estimates (as RBF networks), it is interesting to know how the number of available learning samples is near or not to this theoretical minimum number of samples. In the case where the number of available samples is too small, preprocessing for dimension reduction can be used.

### 1.5.2 Fractal dimension

A classifier is designed for input samples in dimension $d$. This samples distribution can have intrinsically less than $d$ degrees of freedom. The intrinsic dimension lower or equal to $d$ is the dimension of the submanifold structure of the data [18]. The fractal dimension give a local measure of the data dimension [25]. Numerous definitions of the fractal dimension have been proposed [16]. We consider here the similarity dimension. A surface is stated fractal according its self-similarity. A finite data set is self-similar if A is the union of $N_r$ non-overlapping copies of itself. Each copy is similar to A scaled down by a ratio $r$. The fractal dimension $D$ of A is computed by :

$$\begin{aligned} 1 &= N_r \times r^D \\ D &= \frac{log(N_r)}{log(1/r)} \end{aligned}$$

In practice, $D$ is the slope of a regression line. $r$ is the size of the hypercubes which map all the dataset. $N_r$ is the number of hypercubes (size $r$) containing data. the algorithm used here is called the "box counting" algorithm. More details about this methods are available in the R3-B4-P report.

### 1.5.3 Inertia and dispersion

Inertia is a classical measure for the variance of high dimensional data.
We distinguish :

- the global inertia, which is computed over the whole database -$I_G$-,
- the within-class inertia, which is the weighted sum of the inertia computed on each class (the weight is the a priori probability of each class) -$I_W$-,
- the between-class inertia, which is the inertia computed on the centers of gravity of each class -$I_B$-.

Let us consider a database of $N$ samples, with $c$ classes, each feature having been centered and normalized. For the class $\omega_i$, the number of samples is $N_i$ and the center of gravity is $\vec{g}_i$. We have :

$$\begin{aligned} I_G &= \frac{1}{N}\sum_{i=1}^{N} ||\vec{x}(i)||^2 \\ I_{\omega_i} &= \frac{1}{N_i}\sum_{l=1}^{l=N_i} ||\vec{x}(l) - \vec{g}_i||^2;\ \forall \vec{x}(l)\,|\,class(\vec{x}(l)) = \omega_i \\ I_W &= \frac{1}{N}\sum_{i=1}^{i=c} N_i I_{\omega_i} \\ I_B &= \frac{1}{N}\sum_{i=1}^{i=c} N_i ||\vec{g}_i||^2 \end{aligned}$$

where $||.||^2$ is the square of the Euclidean norm: $||\vec{x}||^2 = \vec{x}.\vec{x}^t$.

The classification performances depend on the discrimination power of the features. For the databases, the overlapping between the classes is more or less important. We can measure discrimination or overlapping index. Classical measures for discrimination or separability are the Fisher criteria (FC) and the divergence or the Bhattaharyya distance [24].

We have used here the Fisher criteria and simple measures for the dispersion as the mean dispersion of class $\omega_i$ in class $\omega_j$ by (the notations are the same as here above) :

$$FC = \frac{I_B}{I_W} \tag{1.1}$$

$$Dispersion(i,j) = \frac{||\vec{g}_i - \vec{g}_j||}{\sqrt{I_{\omega_j}}} \tag{1.2}$$

The discrimination is better if the Fisher (FC) criterion is large. If the dispersion measure between two classes is large, these classes are well separated, the between-class distance is more larger than the mean dispersion of the classes. If this measure is close to or lower than one, the classes are very overlapped, but, that doesn't necessary mean an important confusion between these two classes from the classification point of view. For example, it will be the case for multimodal classes or very elongated clusters. It is thus necessary to complete this measure with a reliable estimate of the Bayesian confusion matrix. For this, a simple classification test with the k_NN classifier as best practical reference for the Bayes error can be used in order to evaluate the confusion between the classes.

### 1.5.4 Confusion matrix and confidence intervals

A classifier is always defined by its discriminant function $f$ which divides the $d$-dimensional space into as many regions as there are classes. If there are $c$ classes $\omega_i$, $1 \leq i \leq c$, the discriminant function may also be expressed by the $c$ *class indicating functions* of the classifier $f_i$, where $f_i(u) = 1$ if $f(u) = \omega_i$ and $f_i(u) = 0$ otherwise.

The most useful way to illustrate the performances of a classifier for a given problem is to provide its confusion matrix $C(f)$. Each line of this confusion matrix is dedicated to a particular class; for a given line each element provides the probability (expressed here in percent) for patterns of this class to be attributed to any other or to the original class. We thus have $C_{ij} = 100\, Pr\{\omega_j \text{ assigned when } \omega_i \text{ is true}\}$:

$$C_{ij} = 100 \int p(u|\omega_i) f_j(u) du \tag{1.3}$$

where $p(u|\omega_i)$ is the probability density function of the points belonging to class $\omega_i$. The best confusion matrix is the one corresponding to the Bayesian classifier (minimal attainable classification error $Eo$). Table 1.1 presents the confusion matrix of an hypothetic classifier on a 3-class problem. The classifier performance may also be expressed by the averaged classification error

$$E(f) = \sum_{i=1}^{c} P_i (\sum_{j \neq i} C_{ij}(f)) \tag{1.4}$$

where $P_i$ is the a priori probability of class $\omega_i$.

| *Class* | *0* | *1* | *2* |
|---|---|---|---|
| *0* | 96.6 | 3.4 | 0.0 |
| *1* | 2.5 | 97.5 | 0.0 |
| *2* | 0.0 | 0.0 | 100.0 |

Table 1.1: A confusion matrix.

In the field of classifier learning from examples, the statistics of the problem ($p(u|\omega_i)$ and $P_i$) are never known and the exact confusion matrix of a classifier $C(f)$ can only be estimated over a set of patterns picked up in the database of available samples (the *test set*). This estimate is called the "apparent" confusion $\hat{C}(f)$ in opposition to the exact confusion $C(f)$ of the classifier which is never known. We speak in this case of *error counting methods* and the apparent confusion matrix is obtained by:

$$\hat{C}_{ij} = \frac{100}{N_i} \sum_{k=1}^{N_i} f_j(x(k)) \tag{1.5}$$

where $x(k), 1 \leq k \leq N_i$ are the point of the testset belonging to class $\omega_i$.

From this matrix, the apparent mean classification error $\hat{E}(f)$ is computed by the same method as here above, using for $P_i$ the values estimated from the testset: $\hat{P}_i = N_i(test)/N(test)$.

In practice, the amount of available samples is always finite and is frequently smaller than one would like. It is thus important to utilize at best the amount of available samples for a good design of the classifier and in order to have sufficient confidence in the performance prediction. In [13] Fukunaga shown that the size of the learnset alone account for the degradation in a classifier performance, while the size of the testset dominate the variance of the error estimate.

The most important error-counting methods used for performances estimation are detailed here below.

- *The Resubstitution method*
  The procedure which consists in estimating the classifier and testing its performances from the same data set is called Resubstitution in statistics. It is now quite well known that performances computed with this method are optimistically biased: the nice performance figures on the design set do not extend to independent test sets: it is the overfitting problem.

- *The Holdout method*
  Since the bias of the resubstitution method has been discovered, cross-validation methods have been proposed. They are all based on the same principle: there should be no common data in the learning and in the test sets. In this family is the Holdout, which builds one partition in the set of available patterns into two mutually exclusive subsets (the *learnset* to build the classifier and the other one to test it : the *testset*). In order to make the result less dependent on the partition, one can average several Holdout results, by building several partitions (randomly, or exhaustively drawn); this gives the Averaged Holdout method.

  Since a classifier designed on the entire data set will, on average, perform better than a classifier designed on only a part of the data; this method has a definite tendency to over-estimate the actual error-rate.

- *The Leave-One-Out method*
  This method has been used to compute the confusion matrix of the k_NN classifier provided in this report. The Totally Averaged Leave-One-Out method may be seen as an application of the powerful Jackknife principle to confusion evaluation. In this approach, if $N$ samples are available, $N$ partitions are formed by leaving one single pattern for testing, and using the remaining $N-1$ to build the classifier. The $N$ performance results obtained this way are then averaged. It has been proven that the Leave-One-Out method, like the Holdout, gives in fact an upper bound of the error probabilities, but the estimate is much better, since the learnset sizes are the size of the databases less one. Since the resubstitution method gives a lower bound of these probabilities, the true performance lies in principle in between. The Totally Averaged Leave-One-Out method is a particular case of the Leave-k-Out method explained below.

- *The Leave-k-Out (rotation or m-fold cross validation) method*
  This method is a compromise between the Holdout and the Leave-One-Out estimate. In this case, $m = N/k$ different partitions are formed by leaving $k$ patterns for testing, and using the remaining $N-k$ to build the classifier. The $m$ performance results obtained this way are then averaged. It is clear that when $k = 1$, this method reduces to the Leave-One-Out, whereas when $k = N/2$ it reduces essentially to the Holdout method where the roles of the learn and test sets are interchanged. The rotation method reduces both the bias inherent to the Holdout method and the computational burden of the Leave-One-Out.

Many supervised learning and classification related papers give comparative experiment results by simply ranking $p$ preselected classifiers (designed on a single learning set) on the basis of their respective mean prediction errors on a Holdout testset. This is not very reliable on the statistical point of view, because we have no idea of the variability of these performance estimation (nothing ensures that the same ranking will be obtained for a different Holdout partition of the database). The neural network community begins now to be more interested by statistics and some recent papers have considered this problem: the statistical analysis of comparative experiments has been addressed in [23] and the problem of ranking methods by significance testing in [10].

The most easy way to give some idea of the variability of a performance measure is by the mean of 95% confidence intervals (this method has also been presented in [5] and [8]). For example, if the error rate is obtained by counting $T$ errors in a testset of size $N$, then $T$ is a $binomial(T, N)$ distribution and the standard error of $\hat{p} = T/N$ is $\sqrt{\hat{p}(1-\hat{p})/N}$. This can be quoted in terms of error bars; the usual 95% confidence interval being approximatively plus and minus twice the standard error. These error bars may be computed either individually for each term of the confusion matrix, either for the global error. So, for a given apparent confusion matrix provided in percent, each element may be seen as a random variable $C_{lk}$ having the following 95% confidence interval:

$$\hat{C}_{lk} - 1.96\sqrt{\frac{\hat{C}_{lk}(100-\hat{C}_{lk})}{N_l}} \leq C_{lk} \leq \hat{C}_{lk} + 1.96\sqrt{\frac{\hat{C}_{lk}(100-\hat{C}_{lk})}{N_l}} \qquad (1.6)$$

where $\hat{C}_{lk}$ is the value (in percent) of the considered element of the confusion matrix (line $l$, column $k$) and $N_l$ is the number of points of the testset belonging to class $\omega_l$.

The same argument may be applied to obtain confidence intervals on the apparent mean classification error $\hat{E}(f)$. We have in this case:

$$\hat{E} - 1.96\sqrt{\frac{\hat{E}(100-\hat{E})}{N}} \leq E \leq \hat{E} + 1.96\sqrt{\frac{\hat{E}(100-\hat{E})}{N}} \tag{1.7}$$

# Chapter 2

# The ARTIFICIAL databases

## 2.1 The *Gaussian* database

### 2.1.1 Aim

Benchmarking studies of the classifier behavior for different dimensionalities of the input vectors, for heavy overlapped distributions and for non linear separability.

### 2.1.2 Description

- A set of seven databases corresponding to the same problem, but with dimensionality ranging from 2 to 8.
- The class 0 is represented by a multivariate normal distribution with zero mean and standard deviation egal to 1 in all dimensions, and the class 1 by a normal distribution with zero mean and standard deviation egal to 2 in all dimensions. There are 5000 patterns, 2500 in each class.
- The files containing these databases are named '*gauss_iD.dat*', where i corresponds to the dimensionality. A graphical representation of the two_dimensional database '*gauss_2D.dat*' is provided in file '*gauss_2D.ps*' (figure 2.1).
- In order to test only the influence of a dimension increase, the databases were generated in the same way for all dimensions. The vectors presentation order is thus the same and for a given vector, all the shared attributes in the 7 databases are the same. To give an example, the last vector of the 4 first databases are listed here above:

  '*gauss_2D.dat*':
  -0.428211 -0.881960 0
  '*gauss_3D.dat*':
  -0.428211 -0.881960 -1.558018 0
  '*gauss_4D.dat*':
  -0.428211 -0.881960 -1.558018 -1.368642 0
  '*gauss_5D.dat*':
  -0.428211 -0.881960 -1.558018 -1.368642 0.756191 0

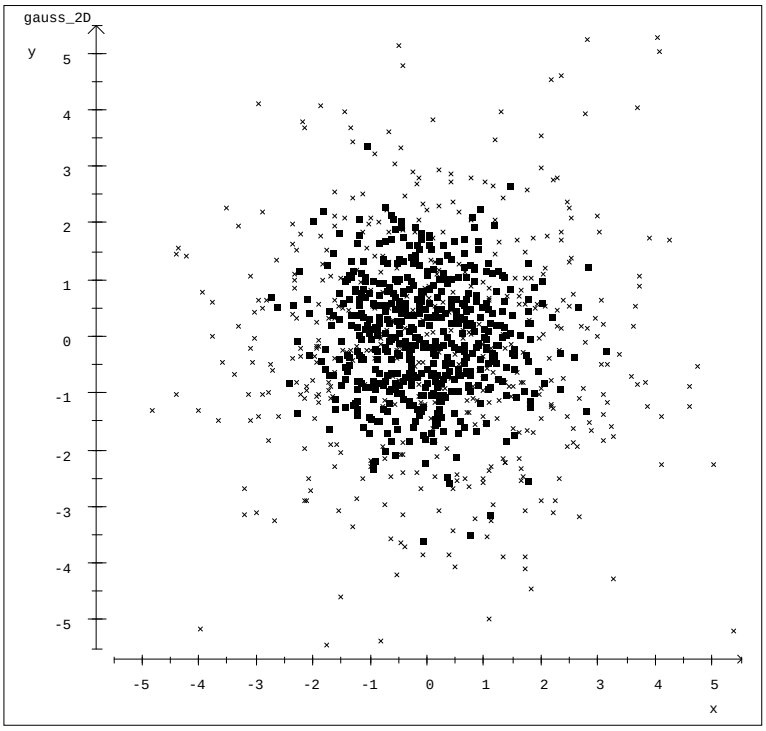


Figure 2.1: The *gauss_2D* database.

- Equivalent databases where already studied by Kohonen in [17]. In this paper, where these databases are referred to as the "hard task", the performances of three basis types of neural-like networks (Backpropagation network, Boltzmann machine and Learning Vector Quantization) are evaluated and compared to the theoretical limit.

### 2.1.3 Theoretical Bayes confusion

For this kind of distributions, the theoretical Bayes boundary is an hypersphere with a radius depending on the database dimensionality $d$ :

$$R = \sqrt{\frac{8}{3} ln(2^d)}\,, \tag{2.1}$$

where $ln$ is the natural logarithm.

The integral of a gaussian probability density function into an hypersphere of radius $R$ is egal to

$$I = \Gamma(\frac{d}{2}, \frac{R^2}{2\sigma^2}), \tag{2.2}$$

where $\Gamma(a, x)$ is the value of the incomplete Gamma function of $a$, integrated to limit $x$.

It is thus very easy to compute the element of the theoretical confusion matrix for the different databases :

| *Class* | *0* | *1* |
|---|---|---|
| *0* | $\Gamma(\frac{d}{2}, \frac{R^2}{2\sigma_0^2})$ | $1 - Conf_{00}$ |
| *1* | $\Gamma(\frac{d}{2}, \frac{R^2}{2\sigma_1^2})$ | $1 - Conf_{10}$ |

This gives:

| $Dim$ | $R$ | $Conf_{00}$ | $Conf_{01}$ | $Conf_{10}$ | $Conf_{11}$ | $Err_{tot}$ |
|---|---|---|---|---|---|---|
| 2 | 1.9227 | 84.25 | 15.75 | 37.00 | 63.00 | 26.37 |
| 3 | 2.3550 | 86.41 | 13.59 | 29.13 | 70.87 | 21.36 |
| 4 | 2.7191 | 88.35 | 11.65 | 23.64 | 76.36 | 17.64 |
| 5 | 3.0401 | 90.02 | 09.98 | 19.53 | 80.47 | 14.75 |
| 6 | 3.3302 | 91.44 | 08.56 | 16.32 | 83.68 | 12.44 |
| 7 | 3.5970 | 92.64 | 07.36 | 13.75 | 86.25 | 10.55 |
| 8 | 3.8454 | 93.66 | 06.34 | 11.66 | 88.34 | 09.00 |

### 2.1.4 Confusion obtained with the k_NN classifier

The confusion matrix was obtained with a Leave_One_Out error counting method and $k$ was set in order to reach the minimum total error.

| $Dim$ | $k$ | $Conf_{00}$ | $Conf_{01}$ | $Conf_{10}$ | $Conf_{11}$ | $Err_{tot}$ |
|---|---|---|---|---|---|---|
| 2 | 33 | 84.4 | 15.6 | 39.1 | 60.9 | 27.3 |
| 3 | 35 | 87.4 | 12.6 | 31.8 | 68.2 | 22.2 |
| 4 | 23 | 90.9 | 09.1 | 29.7 | 70.3 | 19.4 |
| 5 | 13 | 93.1 | 06.9 | 28.6 | 71.4 | 17.7 |
| 6 | 11 | 95.8 | 04.2 | 29.4 | 70.6 | 16.8 |
| 7 | 7 | 96.5 | 03.5 | 28.4 | 71.6 | 15.9 |
| 8 | 5 | 97.3 | 02.7 | 29.6 | 70.4 | 16.1 |

From this table, one can see that, for the k_NN classifier, when the input dimension increases, the performances improve at a rate significantly less than the rate improvement of the theoretical limit. Data of this table are plotted at figure 2.2. This increasing difference between the theoretical errors and the practical estimates is due to the fact that the number of patterns being egal for each dimensionality (5000), the size of the database versus the dimensionality is decreasing while the input dimension increases. This phenomenon is explained in the first axis A report [6].

### 2.1.5 Performances reported by Kohonen

- Multi-Layer Perceptron (BackPropagation)

  A two-layer network was used; this is sufficient to form convex decision regions, which is the case here. A learning rate of 0.01 with a momentum coefficient of 0.9 were chosen. Eight nodes in the input layer, with a number of inputs egal to the dimensionality of the input vectors. Two nodes in the output layer (number

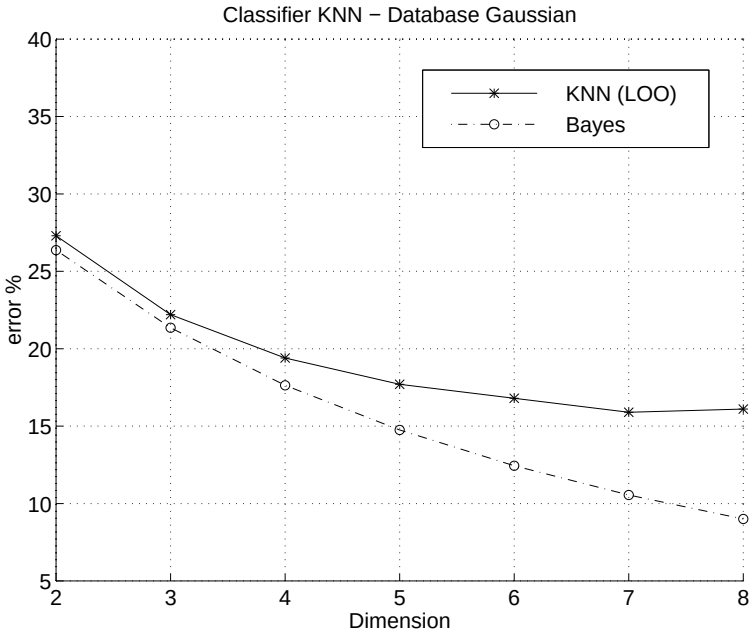


Figure 2.2: Comparison between the theoretical recognition rate and the experimental error rate obtained by the best KNN classifier (error counting by a Leave-One-Out method).

of classes). One bias weight for each node.
There was thus a total of $8d+26$ weights (where d is the dimensionality of input).
The results are shown in the table below.

- Binary Boltzmann Machine

  In each dimension, the input range was splitted into 20 subranges. There were thus 20 input units for each dimension, all of them being set 'off' except the one which was associated with the subrange containing the input value of the dimension. In addition, there were 2 hidden and 2 output units. This corresponds to a number of weights equal to $80d + 3$ (where d is the dimensionality of input). The results are shown in the table here below. Reaching the minimal error level required about $4,500,000$ samples !

- Learning Vector Quantization

  The number of processing units was chosen to be $5d$, thus resulting in $5d^2$ weights. The LVQ1 algorithm was used with the Euclidean metric and the codebook initialization was ensured by k-means. The learning was done with alpha decreasing from 0.01 to 0 over $100,000$ samples. The results are shown on the table here above.

The results reported in the paper [17] only concern the total error rate.
They are:

| Dim | MLP(BP) | BM | LVQ1 | Theoretical limit |
|---|---|---|---|---|
| 2 | 26.3 | 26.5 | 26.5 | 26.37 |
| 3 | 21.5 | 21.6 | 21.8 | 21.36 |
| 4 | 19.4 | 18.0 | 18.8 | 17.64 |
| 5 | 19.5 | 15.2 | 16.9 | 14.75 |
| 6 | 20.7 | 12.7 | 15.3 | 12.44 |
| 7 | 16.7 | 11.0 | 14.5 | 10.55 |
| 8 | 18.9 | 09.4 | 13.4 | 09.00 |

## 2.2 The *Clouds* database

### 2.2.1 Aim

Study of the classifier behavior for heavy intersection of the class distributions and for high degree of nonlinearity of the class boundaries.

### 2.2.2 Description

- Bidimentional distributions, two classes.
  5000 patterns, 2500 in each class (50% in each class).
- Class 0 : sum of three different gaussian distributions with $P_{w_0} = 0.5$ and:

$$p(x|w_0) = \frac{1}{2}\left(\frac{p_1(x)}{2} + \frac{p_2(x)}{2} + p_3(x)\right), \tag{2.3}$$

  with

$$p_j(x) = \frac{1}{2\pi}\frac{1}{\sigma_{jx}\sigma_{jy}} e^{-\left(\frac{(x-m_{jx})^2}{2\,\sigma_{jx}^2} + \frac{(y-m_{jy})^2}{2\,\sigma_{jy}^2}\right)} \tag{2.4}$$

  where $m_{jx}$ and $m_{jy}$ are the $x$ and $y$ means of the $j$'s distribution while $\sigma_{jx}$, $\sigma_{jy}$ are their $x$ and $y$ standard deviations:

| $j$ | $\sigma_{jx}$ | $\sigma_{jy}$ | $m_{jx}$ | $m_{jy}$ |
|---|---|---|---|---|
| 1 | 0.2 | 0.2 | 0.0 | 0.0 |
| 2 | 0.2 | 0.2 | 0.0 | 2.0 |
| 3 | 0.2 | 1.0 | 2.0 | 1.0 |

- Class 1 : a single normal distributions with $P_{w_1} = 0.5$ and:

$$p(x|w_1) = \frac{1}{2\pi} e^{-\frac{x^2+y^2}{2}} \tag{2.5}$$

- Problems of the same type were already studied in: [4] and [3]

  The graphical representation of this dataset is provided in file '*clouds.ps*' (figure 2.3).

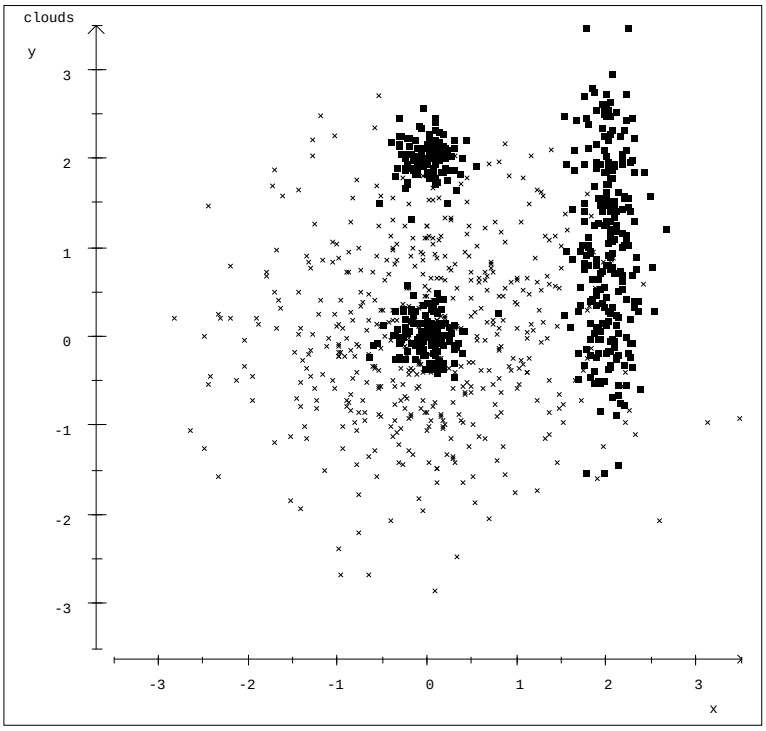


Figure 2.3: The *clouds* database.

### 2.2.3 Theoretical Bayes confusion

| *Class* | *0* | *1* |
|---|---|---|
| *0* | 94.63 | 5.37 |
| *1* | 13.95 | 86.05 |

This gives 9.66% for the average Bayes error (with 0.05% accuracy).

### 2.2.4 Confusion obtained with the k_NN classifier

The confusion matrix was obtained with a Leave_One_Out error counting method and $k$ was set to 5 to reach the minimum total error rate : $10.94 \pm 1.2\%$.

| *Class* | *0* | *1* |
|---|---|---|
| *0* | $92.8 \pm 1.0$ | $7.2 \pm 1.0$ |
| *1* | $14.7 \pm 1.4$ | $85.3 \pm 1.4$ |

## 2.3 The *Concentric* database

### 2.3.1 Aim

Study of the linear separability of the classifier when some classes are nested in other without overlapping.

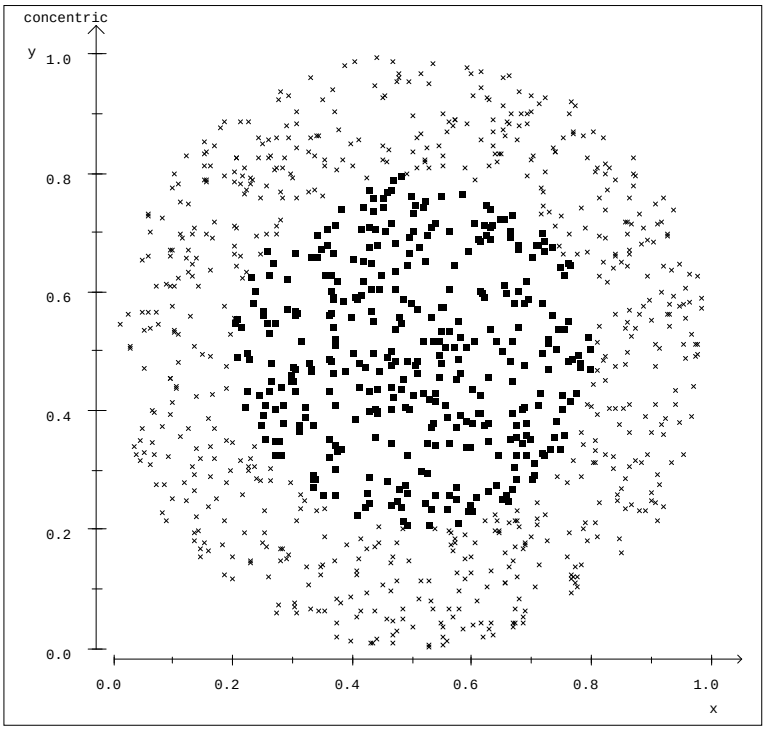


Figure 2.4: The *concentric* database.

### 2.3.2 Description

- Bidimentional uniform concentric circular distributions with two classes.
- The database is entirely contained in the square (0,0), (1,1). There are 2500 instances, 1579 in class 1 (63.2
- The points of class 0 are uniformly distributed into a circle of radius 0.3 centered on (0.5,0.5).
- The points of class 1 are uniformly distributed into a ring centered on (0.5,0.5) with internal and external radius respectively egal to 0.3 and 0.5.

The graphical representation of this dataset is provided in file '*concentric.ps*' (figure 2.4).

### 2.3.3 Theoretical Bayes confusion

| *Class* | *0* | *1* |
|---|---|---|
| *0* | 100.0 | 0.0 |
| *1* | 0.0 | 100.0 |

This gives 0.00% for the average Bayes error (non-overlapping of the classes).

### 2.3.4 Confusion obtained with the k_NN classifier

The confusion matrix was obtained with a Leave_One_Out error counting method and $k$ was set to 7 to reach the minimum mean error rate : $0.96 \pm 0.5\%$.

| *Class* | *0* | *1* |
|---|---|---|
| *0* | $99.1 \pm 0.6$ | $0.9 \pm 0.6$ |
| *1* | $1.0 \pm 0.5$ | $99.0 \pm 0.5$ |

# Chapter 3

# The REAL databases

## 3.1 Landsat image database: *Satimage*

### 3.1.1 Source Information

This database has been taken from the ftp anonymous "*UCI Repository Of Machine Learning Databases and Domain Theories*": [20]. It was in use in the European STATLOG ESPRIT project [21], which involves comparing the performances of machine learning, statistical, and neural network algorithms on data sets from real-world industrial areas including medicine, finance, image analysis, and engineering design.

#### Original source

The original Landsat data for this database was generated from data purchased from NASA by the Australian Centre for Remote Sensing, and used for research at:

> The Centre for Remote Sensing
> University of New South Wales
> Kensington, PO Box 1
> NSW 2033 Australia.

**Past Usage**: [11], [19].

### 3.1.2 Relevant Information

This database was generated from Landsat Multi-Spectral Scanner image data. The Landsat satellite data is one of the many sources of information available for a scene. The interpretation of a scene by integrating spatial data of diverse types and resolutions including multispectral and radar data, maps indicating topography, land use etc. is expected to assume significant importance with the onset of an era characterized by integrative approaches to remote sensing (for example, NASA's Earth Observing System commencing this decade). Existing statistical methods are ill-equipped for handling such diverse data types. Note that this is not true for Landsat MSS data considered in isolation (as in this database). This data satisfies the important requirements of being numerical and at a single resolution, and standard maximum-likelihood classification performs very well. Consequently, for this data, it should be interesting to compare the performance of other methods against the statistical approach.

One frame of Landsat MSS imagery consists of four digital images of the same scene in different spectral bands. Two of these are in the visible region (corresponding approximately to green and red regions of the visible spectrum) and two are in the (near) infra-red. Each pixel is a 8-bit binary word, with 0 corresponding to black and 255 to white. The spatial resolution of a pixel is about 80m x 80m. Each image contains 2340 x 3380 such pixels.

The present database is a (tiny) sub-area of a scene, consisting of 82 x 100 pixels. The binary values were converted to their present ASCII form by Ashwin Srinivasan. The classification for each pixel was performed on the basis of an actual site visit by Ms. Karen Hall, when working for Professor John A. Richards, at the Centre for Remote Sensing at the University of New South Wales, Australia. Conversion to 3x3 neighbourhoods was done by Alistair Sutherland. The initial test and training sets available at the "UCI Repository Of Machine Learning Databases" were concatenated and mixed to obtain this "Satimage" database.

Each line of data corresponds to a 3x3 square neighbourhood of pixels completely contained within the 82x100 sub-area. Each line contains the pixel values in the four spectral bands (converted to ASCII) of each of the 9 pixels in the 3x3 neighbourhood and a number indicating the classification label of the central pixel. The aim is to predict this classification, given the multi-spectral values.

The database contains thus 6435 patterns with 36 attributes (4 spectral bands x 9 pixels in neighbourhood) plus the class label. The attributes are numerical, in the range 0 to 255 (8 bits). The class label is a code for the following classes:

| Number | Class |
|---|---|
| 1 | red soil |
| 2 | cotton crop |
| 3 | grey soil |
| 4 | damp grey soil |
| 5 | soil with vegetation stubble |
| 6 | mixture (all types present) |
| 7 | very damp grey soil |

⚠ There are no examples with class 6 in this dataset : they have all been removed because of doubts about the validity of this class.

The data is given in random order and certain lines of data have been removed so you cannot reconstruct the original image from this dataset.

In each line of data the four spectral values for the top-left pixel are given first followed by the four spectral values for the top-middle pixel and then those for the top-right pixel, and so on with the pixels read out in sequence left-to-right and top-to-bottom. Thus, the four spectral values for the central pixel are given by attributes 17,18,19 and 20. If you like you can use only these four attributes, while ignoring the others. This avoids the problem which arises when a 3x3 neighbourhood straddles a boundary.

### 3.1.3 Summary Statistics

The dynamic of the attributes is in [27-157], with a mean value 83.47 and a standard deviation egal to 17.6. Table here below provides the number of instances in each class

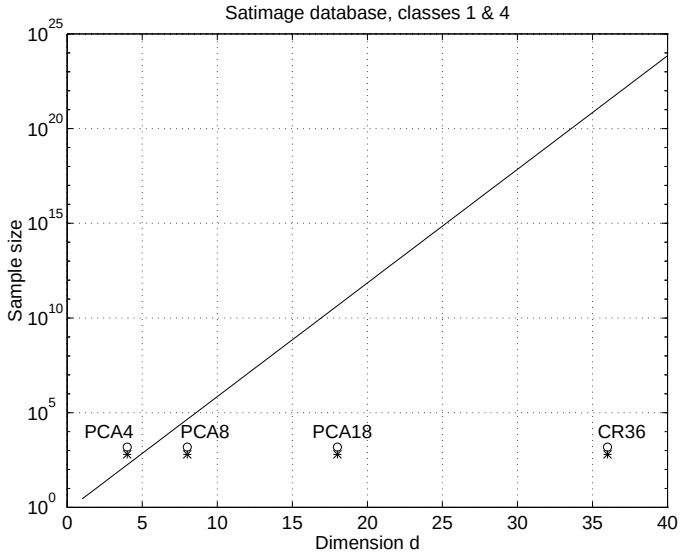


Figure 3.1: Minimum number of points to estimate a *pdf* by kernel functions.

and the estimate of the priors in percents.

| Class | Instances | Percentage |
|---|---|---|
| 1 | 1533 | 23.82 % |
| 2 | 703 | 10.92 % |
| 3 | 1358 | 21.10 % |
| 4 | 626 | 9.73 % |
| 5 | 707 | 10.99 % |
| 7 | 1508 | 23.43 % |

The database resulting from the centering and reduction by attribute of the *Satimage* database is available on the ftp server in the "$/REAL/satimage/satimage\_CR.dat.Z$" file.

Attribute precision: 7 bits.

### 3.1.4 Number of available samples

Figure 3.1 shows that the number of samples is just sufficient for a base in dimension 5 to estimate a probability density function with an error lower than 10%. For greater dimensions, it is a "small" database.

### 3.1.5 Estimate of the fractal dimension

The computation of the fractal dimension by the counting boxes method (figure 3.2) indicates that this distribution of samples would be in a volume of about 8 dimensions. For class one (maximum number of samples), the slope of the curve seems to be greater than 10 (figure 3.3 left) and for class four, the number of points isn't sufficient to identify a straight line on the curve (figure 3.3 right)

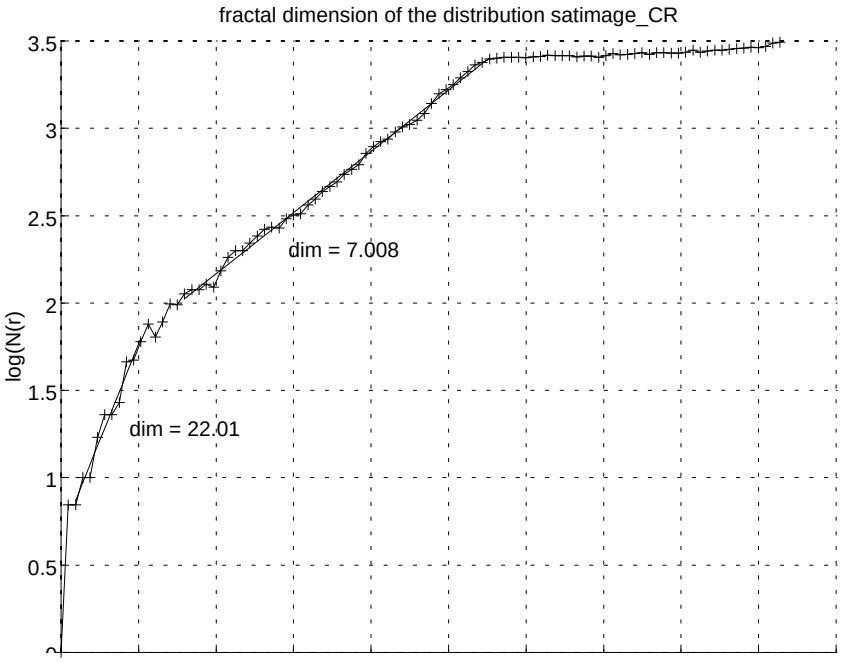


Figure 3.2: Plot of $N(r)$ for the *Satimage* database.

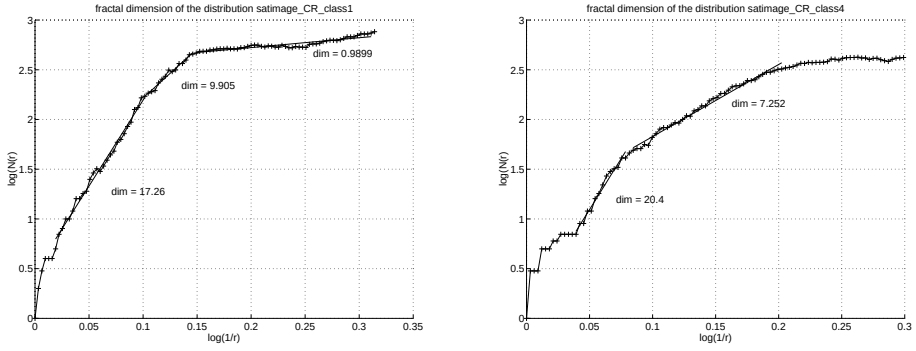


Figure 3.3: Plot of $N(r)$ for class 1 and class 4 of the " Satimage" database.

| *Class* | *1* | *2* | *3* | *4* | *5* | *7* |
|---|---|---|---|---|---|---|
| *1* | $98.1 \pm 0.7$ | $0.2 \pm 0.2$ | $1.1 \pm 0.5$ | $0.1 \pm 0.1$ | $0.5 \pm 0.4$ | $0.0 \pm 0.0$ |
| *2* | $0.0 \pm 0.0$ | $96.5 \pm 1.4$ | $0.1 \pm 0.3$ | $0.7 \pm 0.6$ | $2.0 \pm 1.0$ | $0.7 \pm 0.6$ |
| *3* | $0.5 \pm 0.4$ | $0.1 \pm 0.2$ | $93.4 \pm 1.3$ | $4.6 \pm 1.1$ | $0.0 \pm 0.0$ | $1.4 \pm 0.6$ |
| *4* | $0.0 \pm 0.0$ | $0.8 \pm 0.7$ | $13.7 \pm 2.7$ | $70.6 \pm 3.6$ | $0.8 \pm 0.7$ | $14.1 \pm 2.7$ |
| *5* | $3.1 \pm 1.3$ | $0.8 \pm 0.7$ | $0.1 \pm 0.3$ | $0.8 \pm 0.7$ | $89.7 \pm 2.3$ | $5.5 \pm 1.7$ |
| *7* | $0.0 \pm 0.0$ | $0.1 \pm 0.1$ | $1.9 \pm 0.7$ | $7.3 \pm 1.3$ | $2.0 \pm 0.7$ | $88.7 \pm 1.6$ |

Table 3.1: Confusion matrix with a KNN classifier estimated by a Leave_One_Out method for the *Satimage* database. Averaged error: $8.89 \pm 1.6\%$

| *Class* | *1* | *2* | *3* | *4* | *5* | *7* |
|---|---|---|---|---|---|---|
| *1* | 0.0 | 1.7673 | 2.1275 | 1.5894 | 1.6166 | 2.2870 |
| *2* | 2.4216 | 0.0 | 4.9302 | 4.1477 | 2.3935 | 4.2002 |
| *3* | 1.4792 | 2.5017 | 0.0 | 1.6006 | 2.3436 | 2.8327 |
| *4* | 1.1845 | 2.2559 | 1.7157 | 0.0 | 1.3250 | 1.2788 |
| *5* | 1.7910 | 1.9352 | 3.7342 | 1.9697 | 0.0 | 1.0246 |
| *7* | 1.7495 | 2.3449 | 3.1166 | 1.3127 | 0.7075 | 0.0 |

Table 3.2: Dispersion matrix computed on the *Satimage* database.

### 3.1.6 Confusion matrix obtained with the k_NN classifier

The confusion matrix was obtained with a Leave_One_Out error counting method and $k$ was set to 3 to reach the minimum mean error rate : $8.89 \pm 1.6\%$ (table 3.1).

### 3.1.7 Inertia and dispersion

For the initial database (after centering and reduction), the between-class inertia is 24.03, the within-class inertia is 11.97, and the Fischer's coefficient is 2.01. The dispersion matrix is given at table 3.2. From this dispersion matrix computed from equation 1.2, we see that class 2 has few overlapping with classes 3, 4 and 7. This is confirmed by the confusion matrix. Moreover classes 5 and 7 seems to be overlapped, but this is not confirmed by the confusion matrix, these classes can thus be multimodal or very elongated.

### 3.1.8 Result of the Principal Component Analysis

A principal component analysis was performed on the *Satimage* database. The database resulting from this preprocessing is on the ftp server in the "*/REAL/satimage/satimage_PCA.dat.Z*" file. Table (3.3) provides the inertia percentages associated to the eigenvalues corresponding to the 17 first principal component axis. 99 percent of the total database inertia will remain if the 17 first principal components are kept.

Figure 3.6 shows the evolution of the inertia and the 1_NN recognition error, versus the number of the first principal components kept. The inertia increases quickly with the dimension. With 18 features, the percentage of resulted inertia is just greater 99%.

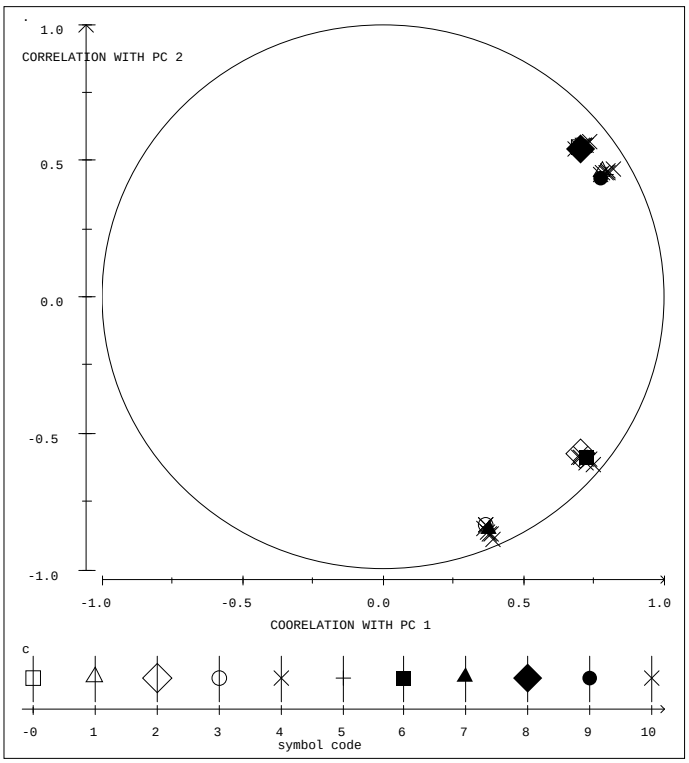


Figure 3.4: The correlation circle of the *satimage* database.

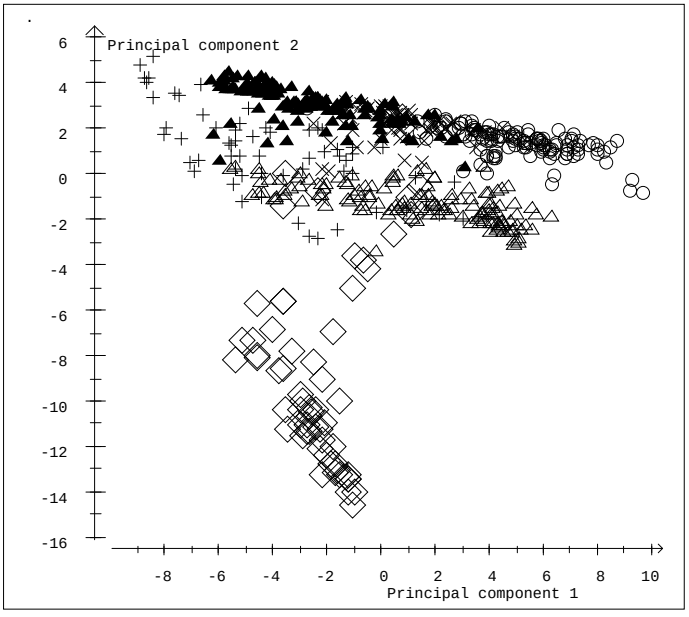


Figure 3.5: The projection of the *Satimage* database on its two first principal components.

| Eigen value | Value | Inertia percentage | Cumulated inertia |
|---|---|---|---|
| 1 | 16.3274 | 45.35 | 45.35 |
| 2 | 14.3575 | 39.88 | 85.24 |
| 3 | 1.57658 | 4.38 | 89.61 |
| 4 | 0.88933 | 2.47 | 92.09 |
| 5 | 0.65945 | 1.83 | 93.92 |
| 6 | 0.60908 | 1.69 | 95.61 |
| 7 | 0.37060 | 1.03 | 96.64 |
| 8 | 0.19197 | 0.53 | 97.17 |
| 9 | 0.12981 | 0.36 | 97.53 |
| 10 | 0.12588 | 0.35 | 97.88 |
| 11 | 0.08386 | 0.23 | 98.11 |
| 12 | 0.06657 | 0.18 | 98.30 |
| 13 | 0.06449 | 0.18 | 98.48 |
| 14 | 0.05722 | 0.16 | 98.64 |
| 15 | 0.04557 | 0.13 | 98.77 |
| 16 | 0.04422 | 0.12 | 98.89 |
| 17 | 0.04078 | 0.11 | 99.00 |

Table 3.3: The inertia percentages associated to the 17 first eigenvalues of the *Satimage* database.

This number of dimensions corresponds also to a beginning of a weak increase of the error rate.

The correlation circle (see the *PCA module* chapter in the Task B3 report for more details) corresponding to this database is provided on figure (3.4) and the projection of the database on its two first principal components is given on figure (3.5)

## 3.2 Phonemes recognition database: *Phoneme*

### 3.2.1 Source Information

This database was in use in the European ROARS ESPRIT project [1]. The aim of this project is the development and the implementation of a real time analytical system for French speech recognition.

Dominique VAN CAPPEL
THOMSON-SINTRA,
525 route des Dolines, BP157,
F-06903 Sophia Antipolis Cedex, France

**Past Usage**: [1].

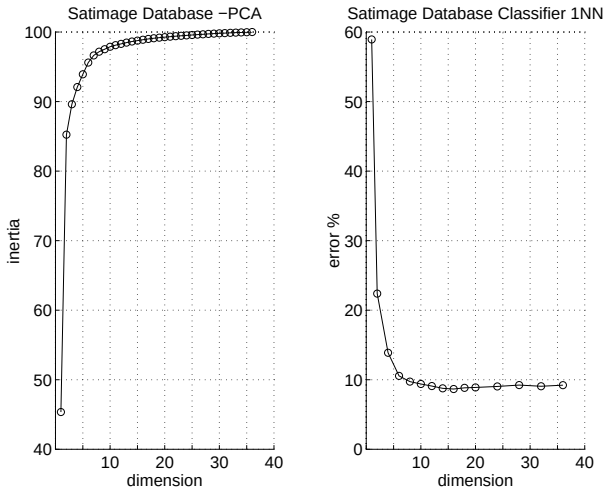


Figure 3.6: Percentage of resulted inertia (left), recognition error with a rule of the first nearest neighbor (right) versus the number of features.

### 3.2.2 Relevant Information

Most of the already existing speech recognition systems are global systems (typically Hidden Markov Models and Time Delay Neural Networks) which recognizes signals and do not really use the speech specificities. On the contrary, analytical systems take into account the articulatory process leading to the different phonemes of a given language, the idea being to deduce the presence of each of the phonetic features from the acoustic observation.

The main difficulty of analitycal systems is to obtain acoustical parameters sufficiently reliable. These acoustical measurements must :

- contain all the information relative to the concerned phonetic feature.
- being speaker independent.
- being context independent.
- being more or less robust to noise.

The primary acoustical observation is always voluminous (spectrum x $N$ different observation moments) and classification cannot been processed directly.

In ROARS, the initial database is provided by a cochlear spectra, which may be seen as the output of a filters bank having a constant $\Delta F/F_0$, where the central frequencies are distributed on a logarithmic scale (MEL type) to simulate the frequency answer of the auditory nerves. The filters outputs are taken every 2 or 8 msec (integration on 4 or 16 msec) depending on the type of phoneme observated (stationnary or transitory).

The aim of the present database is to distinguish between nasal and oral vowels. There are thus two different classes: the *Nasals* in Class 0 and the *Orals* in Class 1. This database contains vowels coming from 1809 isolated syllables (for example: pa, ta, pan,...). Five different attributes were chosen to characterize each vowel: they are the amplitudes of the five first harmonics $AH_i$, normalized by the total energy $Ene$

(integrated on all the frequencies): $AH_i/Ene$. Each harmonic is signed: positive when it corresponds to a local maximum of the spectrum and negative otherwise.

Three observation moments have been kept for each vowel to obtain 5427 different instances: the observation corresponding to the maximum total energy $Ene$ and the observations taken 8 msec before and 8 msec after the observation corresponding to this maximum total energy.

From these 5427 initial values, 23 instances for which the amplitude of the 5 first harmonics was zero were removed, leading to the 5404 instances of the present database.

### 3.2.3 Summary Statistics

- Table here below provides for each attribute of the database the dynamic (Min and Max values), the mean value and the standard deviation.

| Attribute | Min | Max | Mean | Standard deviation |
|---|---|---|---|---|
| 1 | -1.70 | 4.11 | 0.82 | 0.86 |
| 2 | -1.33 | 4.38 | 1.26 | 0.85 |
| 3 | -1.82 | 3.20 | 0.76 | 0.93 |
| 4 | -1.58 | 2.83 | 0.40 | 0.80 |
| 5 | -1.28 | 2.72 | 0.08 | 0.58 |

- Class Distribution: number of instances and percentage per class:

| Class | Instances | Percentage |
|---|---|---|
| 0 | 3818 | 70.65 % |
| 1 | 1586 | 29.35 % |

- Correlation matrix:

$$\begin{pmatrix} 1.00 & -0.10 & -0.32 & -0.19 & -0.05 \\ -0.10 & 1.00 & -0.25 & -0.21 & -0.07 \\ -0.32 & -0.25 & 1.00 & 0.02 & 0.01 \\ -0.19 & -0.21 & 0.02 & 1.00 & -0.04 \\ -0.05 & -0.07 & 0.01 & -0.04 & 1.00 \end{pmatrix}$$

- Attribute precision: 15 bits.

- The database resulting from the centering and reduction by attribute of the *Phoneme* database is on the ftp server in the "$/REAL/phoneme/phoneme\_CR.dat$" file.

### 3.2.4 Number of available samples

This database is composed of two classes in 5 dimensions. There are 5404 patterns; 3818 for class zero and 1586 for class one. Figure 3.7 shows that the number of samples is just sufficient for the database dimension (5).

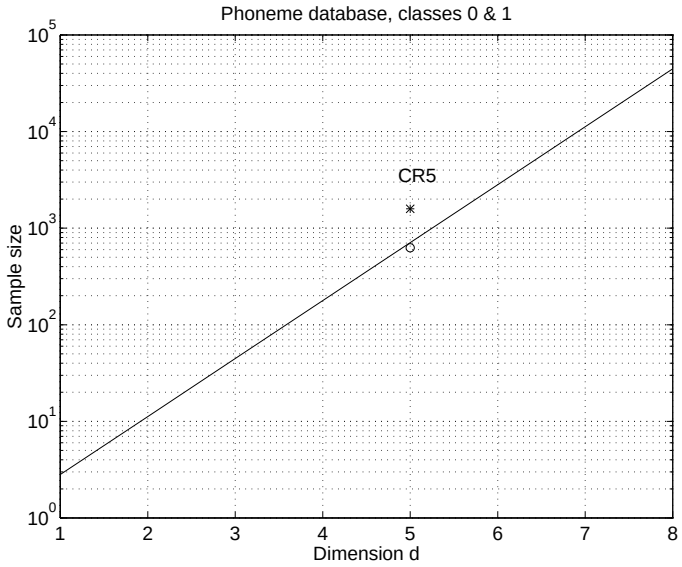


Figure 3.7: Minimum number of points to estimate a *pdf* by kernel functions, and the location of the two classes for the *Phoneme* database.

### 3.2.5 Estimate of the fractal dimension

The computation of the fractal dimension (figure 3.8) indicates that this distribution of samples would be in a volume in 5 dimensions even if each class is viewed in space with a lower number of dimensions (figure 3.9).

### 3.2.6 Confusion matrix obtained with the k_NN classifier

The confusion matrix was obtained with a Leave_One_Out error counting method and $k$ was set to 1 to reach the minimum mean error rate : $8.97 \pm 1.1\%$.

| *Class* | *0* | *1* |
|---|---|---|
| *0* | $95.0 \pm 0.7$ | $5.0 \pm 0.7$ |
| *1* | $18.5 \pm 1.9$ | $81.5 \pm 1.9$ |

With $k$ set to 1, this result is a bit optimistic because of the fact that the database is composed of the same phonemes taken at 3 differents moments. Setting $k$=20, will permit to avoid this influence and will provide more realistic results:

| *Class* | *0* | *1* |
|---|---|---|
| *0* | 91.40 | 8.60 |
| *1* | 27.80 | 72.20 |

In this case, the mean error rate is: 14.2 %.

### 3.2.7 Inertia and dispersion

The between-class inertia is 0.35, the within-class inertia is 4.64, and the Fischer's coefficient is 0.0756. The dispersion matrix is given at table 3.4. We see that with these parameters, there is dispersion between the two classes, and these two classes are overlapped.

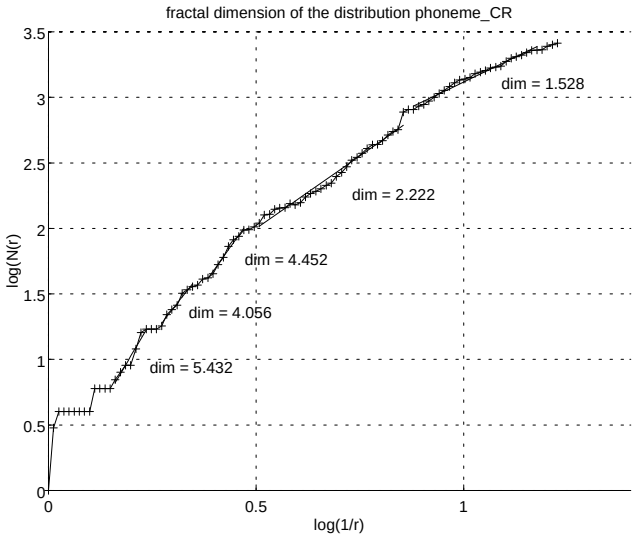


Figure 3.8: Plot of $N(r)$ for the *Phoneme* database.

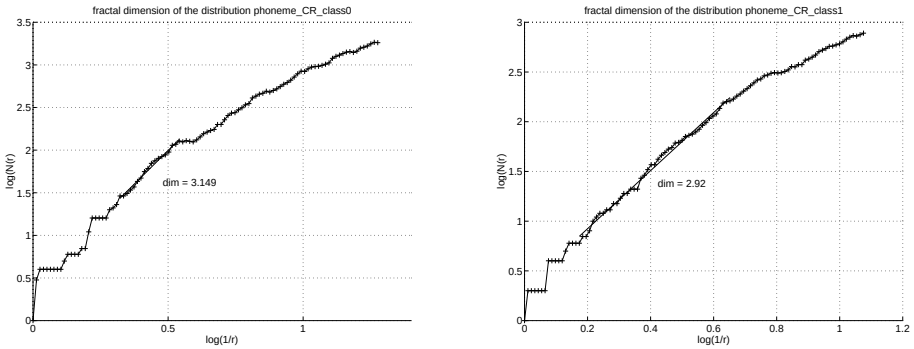


Figure 3.9: Plot of $N(r)$ for class 0 and class 1 of the *Phoneme* database.

| *Class* | *0* | *1* |
|---|---|---|
| *0* | 0.0 | 0.62 |
| *1* | 0.60 | 0.0 |

Table 3.4: Dispersion matrix computed on the *Phoneme* database.

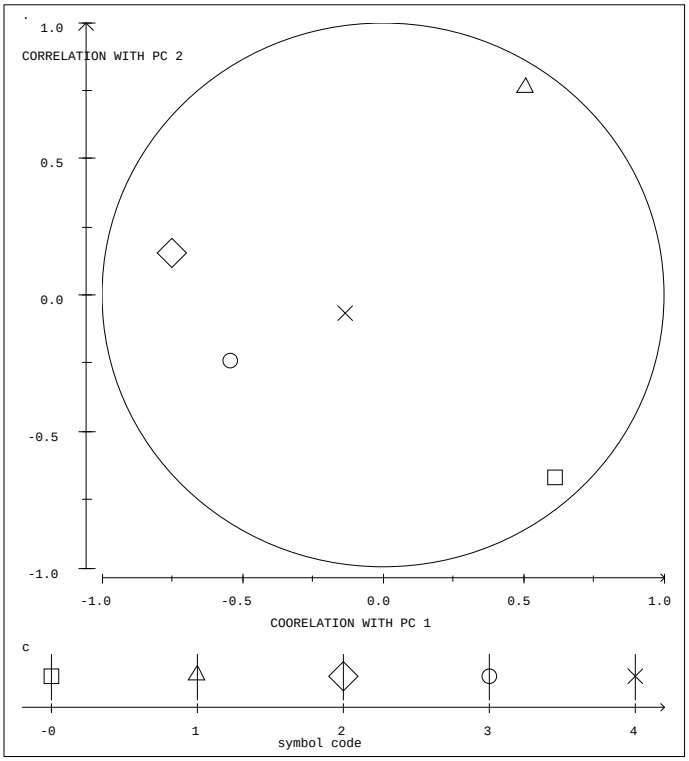


Figure 3.10: The correlation circle of the *phoneme* database.

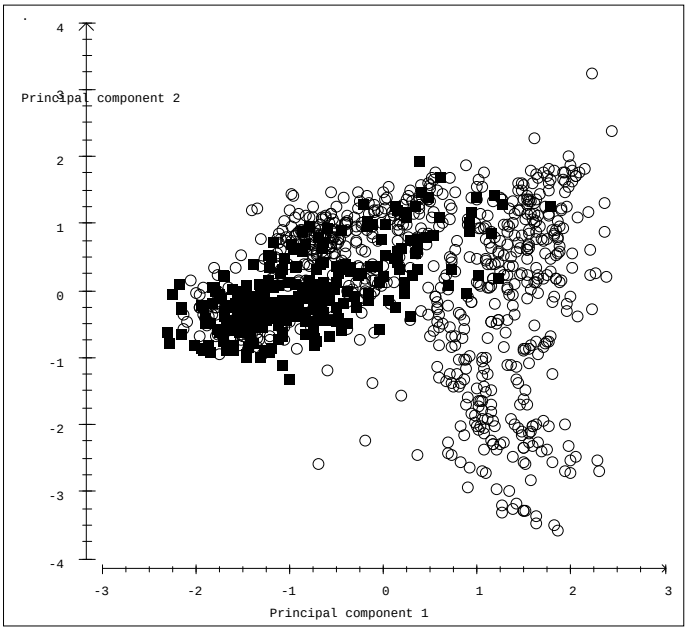


Figure 3.11: The projection of the *Phoneme* database on its two first principal components.

| Eigen value | Value | Inertia percentage | Cumulated inertia |
|---|---|---|---|
| 1 | 1.46498 | 29.23 | 29.23 |
| 2 | 1.10954 | 22.19 | 51.49 |
| 3 | 1.02849 | 20.57 | 72.06 |
| 4 | 0.94175 | 18.83 | 90.89 |
| 5 | 0.45524 | 9.10 | 100.00 |

Table 3.5: The inertia percentages associated to the 5 eigenvalues of the *Phoneme* database.

### 3.2.8 Result of the Principal Component Analysis

A principal component analysis was performed on the *Phoneme* database. The database resulting from this preprocessing is on the ftp server in the "*/REAL/phoneme/phoneme_PCA.dat.Z*" file. Table (3.5) provides the inertia percentages associated to the eigenvalues corresponding to the principal component axis. It is clear that any dimensionality reduction based on PCA would lead to an important loss of pertinent data.

The correlation circle (see the *PCA module* chapter in the Task B3 report for more details) corresponding to this database is provided on figure (3.10) and the projection of the database on its two first principal components is given on figure (3.11)

## 3.3 Iris plants database: *Iris*

### 3.3.1 Source Information

This database has been taken from the ftp anonymous "*UCI Repository Of Machine Learning Databases and Domain Theories*": [20].

**Original source**

Anderson, E. (1935) "The Irises of the Gaspe Peninsula",
Bulletin of the American Iris Society, 59, 2-5.

**Donor**

Michael Marshall
(MARSHALL%PLU@io.arc.nasa.gov)

### 3.3.2 Relevant Information

- **Past Usage**: [12], [9], [7], [14], ...
- This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper [12] is a classic in the field and is referenced frequently to this day (See [9] for example).

- The data set contains 3 classes of 50 instances each, with 4 numeric, predictive attributes. Each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.
- This is an exceedingly simple domain. Very low misclassification rates are obtained in [7] (0% misclassification for the class 0) and in [14].
- Attribute Information:
  1. sepal length in cm
  2. sepal width in cm
  3. petal length in cm
  4. petal width in cm
  5. class:

     Iris Setosa (class 0)
     Iris Versicolour (class 1)
     Iris Virginica (class 2)

### 3.3.3 Examples

5.1 3.5 1.4 0.2 0
4.9 3.0 1.4 0.2 0
4.7 3.2 1.3 0.2 0
4.6 3.1 1.5 0.2 1
5.0 3.6 1.4 0.2 1

### 3.3.4 Summary Statistics

- Table here below provides for each attribute of the database the dynamic (Min and Max values), the mean value and the standard deviation.

| Attribute | Min | Max | Mean | Standard deviation |
|---|---|---|---|---|
| sepal length (1) | 4.3 | 7.9 | 5.84 | 0.83 |
| sepal width (2) | 2.0 | 4.4 | 3.05 | 0.43 |
| petal length (3) | 1.0 | 6.9 | 3.76 | 1.76 |
| petal width (4) | 0.1 | 2.5 | 1.20 | 0.76 |

- Class Distribution: number of instances and percentage per class:

| Class | Instances | Percentage |
|---|---|---|
| 0 | 50 | 33.3 % |
| 1 | 50 | 33.3 % |
| 2 | 50 | 33.3 % |

- Correlation matrix:

$$\begin{pmatrix} 1.00 & -0.11 & 0.87 & 0.82 \\ -0.11 & 1.00 & -0.42 & -0.36 \\ 0.87 & -0.42 & 1.00 & 0.96 \\ 0.82 & -0.36 & 0.96 & 1.00 \end{pmatrix}$$

- Attribute precision: 10 bits.

### 3.3.5 Inertia and dispersion

The between-class inertia is 2.86, the within-class inertia is 1.13, and the Fischer's coefficient is 2.52. The dispersion matrix is given bellow. In this table, the fact that class zero is more separated from the others is confirmed.

| *Class* | *0* | *1* | *2* |
|---|---|---|---|
| *0* | 0.0 | 2.76 | 3.36 |
| *1* | 2.85 | 0.0 | 1.27 |
| *2* | 4.0 | 1.45 | 0.0 |

### 3.3.6 Confusion matrix obtained with the k_NN classifier

The confusion matrix was obtained with a Leave_One_Out error counting method and $k$ was set to 7 to reach the minimum mean error rate : $3.33 \pm 4.0\%$.

| *Class* | *0* | *1* | *2* |
|---|---|---|---|
| *0* | $100.0 \pm 0.0$ | $0.0 \pm 0.0$ | $0.0 \pm 0.0$ |
| *1* | $0.0 \pm 0.0$ | $96.0 \pm 5.4$ | $4.0 \pm 5.4$ |
| *2* | $0.0 \pm 0.0$ | $6.0 \pm 6.6$ | $94.0 \pm 6.6$ |

### 3.3.7 Result of the Principal Component Analysis

A PCA was performed on the *Iris* database. The database resulting from this pre-processing is on the ftp server in the "*/REAL/iris/iris_PCA.dat.Z*" file. Table here below provides the inertia percentages associated to the eigenvalues corresponding to the principal component axis.

| Eigen value | Value | Inertia percentage | Cumulated inertia |
|---|---|---|---|
| 1 | 2.9108200 | 72.8 | 72.8 |
| 2 | 0.9212210 | 23.0 | 95.8 |
| 3 | 0.1473530 | 3.7 | 99.5 |
| 4 | 0.0206077 | 0.5 | 100.0 |

The correlation circle (see the *PCA module* chapter in the Task B3 report for more details) corresponding to this database is provided on figure (3.12) and the projection of the database on its two first principal components is given on figure (3.13)

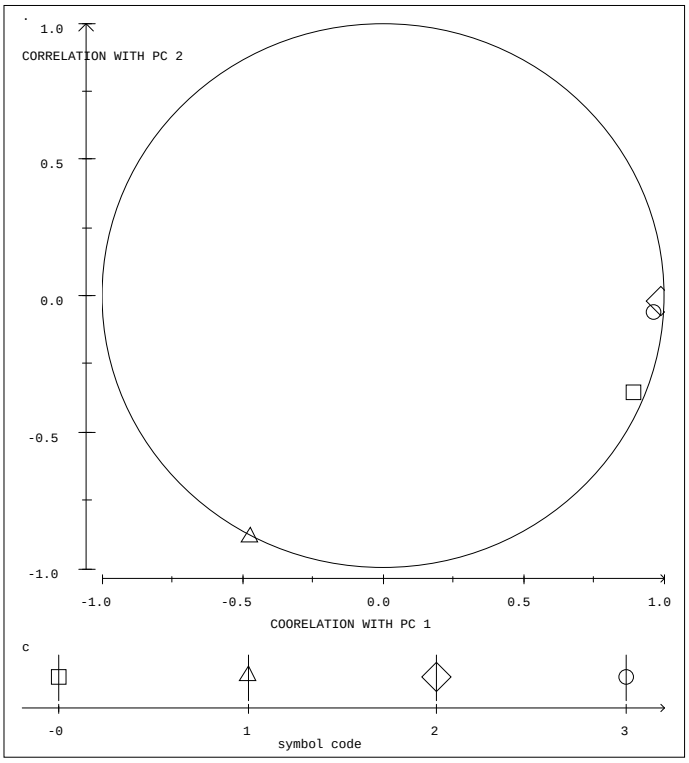


Figure 3.12: The correlation circle of the *Iris* database.

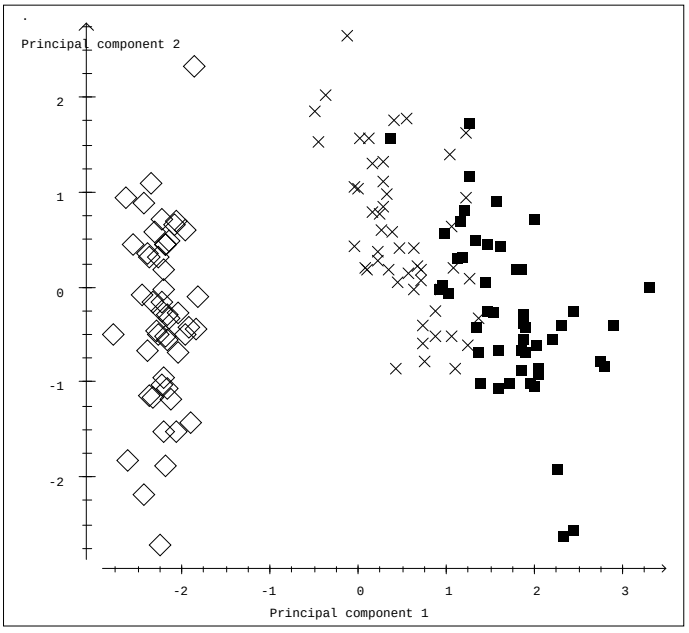


Figure 3.13: The projection of the *Iris* database on its two first principal components.

## 3.4 Texture discrimination database : *Texture*

### 3.4.1 Source Information

This database was generated at the Laboratory of Image Processing and Pattern Recognition's software in the framework of the European Esprit project ELENA No. 6891 and the Esprit working group ATHOS No. 6620.

#### Original source

P. Brodatz "Textures: A Photographic Album for Artists and Designers", Dover Publications Inc., New York, 1966.

#### Past Usage

This database has a private usage at the TIRF laboratory. It has been created in order to study the textures discrimination with high order statistics [15].

### 3.4.2 Relevant Information

A statistical method based on the extraction of fourth order moments for the characterization of natural micro-textures was developed called "fourth order modified moments" (mm4) [15], this method measures the deviation from first-order Gauss-Markov process, for each texture. The features were estimated in four directions to take into account the possible orientations of the textures (0, 45, 90 and 135 degrees). Only correlation between the current pixel, the first neighbourhood and the second neighbourhood are taken into account. This small neighbourhood is adapted to the fine grain property of the textures.

The data set contains 11 classes of 500 instances and each class refers to a type of textures in the Brodatz album. The database dimension is 40 plus one for the class identifier. The 40 attributes were build by the estimation of the following fourth order modified moments in four orientations: 0, 45, 90 and 135 degrees: mm4(000), mm4(001), mm4(002), mm4(011), mm4(012), mm4(022), mm4(111), mm4(112), mm4(122) and mm4(222).

The class label is a code for the following classes:

| Number | Class |
|---|---|
| 2 | Grass lawn (D09) |
| 3 | Pressed calf leather (D24) |
| 4 | Handmade paper (D57) |
| 6 | Raffia looped to a high pile: (D84) |
| 7 | Cotton canvas (D77) |
| 8 | Pigskin (D92) |
| 9 | Beach sand: (D28) |
| 10 | Beach sand (D29) |
| 12 | Oriental straw cloth (D53) |
| 13 | Oriental straw cloth (D78) |
| 14 | Oriental grass fiber cloth (D79) |

### 3.4.3 Summary Statistics

Table here below provides for each attribute of the database the dynamic (Min and Max values), the mean value and the standard deviation.

| Class | Min | Max | Mean | Std dev |
|---|---|---|---|---|
| 1 | -1.4495 | 0.7741 | -1.0983 | 0.2034 |
| 2 | -1.2004 | 0.3297 | -0.5867 | 0.2055 |
| 3 | -1.3099 | 0.3441 | -0.5838 | 0.3135 |
| 4 | -1.1104 | 0.5878 | -0.4046 | 0.2302 |
| 5 | -1.0534 | 0.4387 | -0.3307 | 0.2360 |
| 6 | -1.0029 | 0.4515 | -0.2422 | 0.2225 |
| 7 | -1.2076 | 0.5246 | -0.6026 | 0.2003 |
| 8 | -1.0799 | 0.3980 | -0.4322 | 0.2210 |
| 9 | -1.0570 | 0.4369 | -0.3317 | 0.2361 |
| 10 | -1.2580 | 0.3546 | -0.5978 | 0.3268 |
| 11 | -1.4495 | 0.7741 | -1.0983 | 0.2034 |
| 12 | -1.0831 | 0.3715 | -0.5929 | 0.2056 |
| 13 | -1.1194 | 0.6347 | -0.4019 | 0.3368 |
| 14 | -1.0182 | 0.1573 | -0.6270 | 0.1390 |
| 15 | -0.9435 | 0.1642 | -0.4482 | 0.1952 |
| 16 | -0.9944 | 0.0357 | -0.5763 | 0.1587 |
| 17 | -1.1722 | 0.0201 | -0.7331 | 0.1955 |
| 18 | -1.0174 | 0.1155 | -0.4919 | 0.2335 |
| 19 | -1.0044 | 0.0833 | -0.4727 | 0.2257 |
| 20 | -1.1800 | 0.4392 | -0.4831 | 0.3484 |
| 21 | -1.4495 | 0.7741 | -1.0983 | 0.2034 |
| 22 | -1.2275 | 0.5963 | -0.7363 | 0.2220 |
| 23 | -1.3412 | 0.4464 | -0.7771 | 0.3290 |
| 24 | -1.1774 | 0.6882 | -0.5770 | 0.2646 |
| 25 | -1.1369 | 0.4098 | -0.5085 | 0.2538 |
| 26 | -1.1099 | 0.3725 | -0.4038 | 0.2515 |
| 27 | -1.2393 | 0.6120 | -0.7279 | 0.2278 |
| 28 | -1.1540 | 0.4221 | -0.5863 | 0.2446 |
| 29 | -1.1323 | 0.3916 | -0.5090 | 0.2526 |
| 30 | -1.4224 | 0.4718 | -0.7708 | 0.3264 |
| 31 | -1.4495 | 0.7741 | -1.0983 | 0.2034 |
| 32 | -1.1789 | 0.5647 | -0.6463 | 0.1890 |
| 33 | -1.1473 | 0.6755 | -0.4919 | 0.3304 |
| 34 | -1.1228 | 0.3132 | -0.6435 | 0.1441 |
| 35 | -1.0145 | 0.3396 | -0.4918 | 0.1922 |
| 36 | -1.0298 | 0.1560 | -0.5934 | 0.1704 |
| 37 | -1.2534 | 0.0899 | -0.7795 | 0.1641 |
| 38 | -1.0966 | 0.1944 | -0.5541 | 0.2111 |
| 39 | -1.0765 | 0.2019 | -0.5230 | 0.2015 |
| 40 | -1.2155 | 0.4647 | -0.5677 | 0.3091 |

The dynamic of the attributes is in [-1.45 - 0.775]. The database resulting from the centering and reduction by attribute of the *Texture* database is on the ftp server in the "*/REAL/texture/texture_CR.dat.Z*" file.

### 3.4.4 Number of available samples

Figure 3.14 shows that the number of samples is just sufficient for a base in dimension 4. For greater dimensions, it is a "small" database.

### 3.4.5 Estimate of the fractal dimension

The computation of the fractal dimension (figure 3.15) indicates that this distribution of samples would be in a volume of about 8 dimensions. The same method applied to class 2 and class 10 gives respectively a dimension about to 7 and 3 (figure 3.16). With

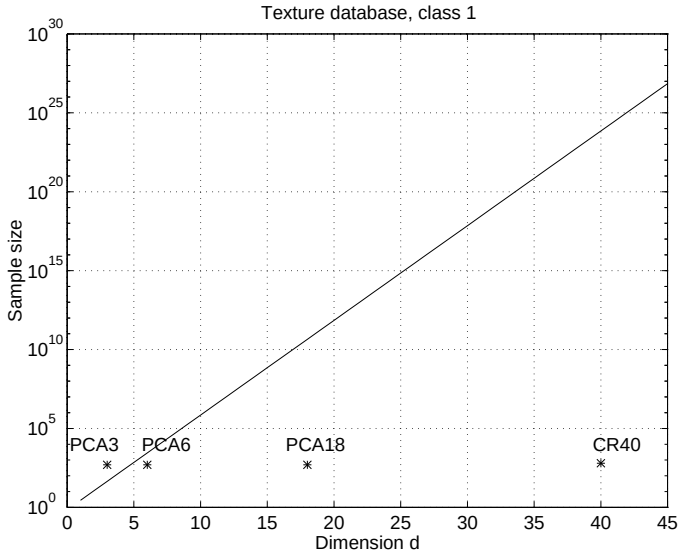


Figure 3.14: Minimum number of points to estimate a *pdf* by kernel functions, and the location of one class for the *Texture* database.

| *Class* | *2* | *3* | *4* | *6* | *7* | *8* | *9* | *10* | *12* | *13* | *14* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *2* | 97.0 | 1.0 | 0.4 | 0.0 | 0.0 | 0.0 | 1.6 | 0.0 | 0.0 | 0.0 | 0.0 |
| *3* | 0.2 | 99.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.4 | 0.0 | 0.0 | 0.0 | 0.4 |
| *4* | 1.0 | 0.0 | 98.8 | 0.0 | 0.0 | 0.0 | 0.2 | 0.0 | 0.0 | 0.0 | 0.0 |
| *6* | 0.0 | 0.0 | 0.0 | 99.4 | 0.0 | 0.0 | 0.0 | 0.6 | 0.0 | 0.0 | 0.0 |
| *7* | 0.0 | 0.0 | 0.0 | 0.0 | 100.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| *8* | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 98.6 | 0.0 | 1.4 | 0.0 | 0.0 | 0.0 |
| *9* | 0.4 | 0.0 | 0.2 | 0.0 | 0.0 | 0.2 | 98.8 | 0.4 | 0.0 | 0.0 | 0.0 |
| *10* | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.4 | 0.0 | 98.6 | 0.0 | 0.0 | 0.0 |
| *12* | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 100.0 | 0.0 | 0.0 |
| *13* | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 99.8 | 0.2 |
| *14* | 0.0 | 0.4 | 0.0 | 0.0 | 0.0 | 0.4 | 0.0 | 0.0 | 0.2 | 0.0 | 99.0 |

Table 3.6: Confusion matrix with a KNN classifier estimated by a Leave-One-Out method for the "Texture" database. Averaged error: 1.00%

this database, it seems that the number of freedom degrees is quite small relatively to the input dimension. This is consistent with the characteristics of the statistical texture features used here. Indeed, more or less complex relations exist between the features according to the samples distribution for a given texture. For two different textures, it isn't necessary the same features which are discriminant and the relations between the features are not necessarily the same.

### 3.4.6 Confusion matrix obtained with the k_NN classifier

The confusion matrix was obtained with a Leave_One_Out error counting method and $k$ was set to 1 to reach the minimum mean error rate : $1.00 \pm 0.8\%$ (table 3.6).

### 3.4.7 Inertia and dispersion

For the initial database (after centering and reduction), the between-class inertia is 29.62, the within-class inertia is 10.39, and the Fischer's coefficient is 2.85. For this

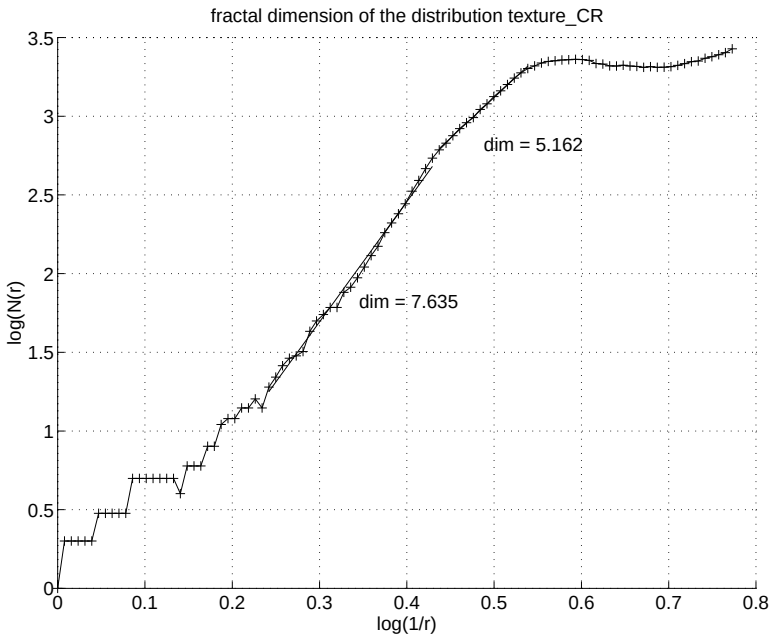


Figure 3.15: Plot of $N(r)$ for the *Texture* database.

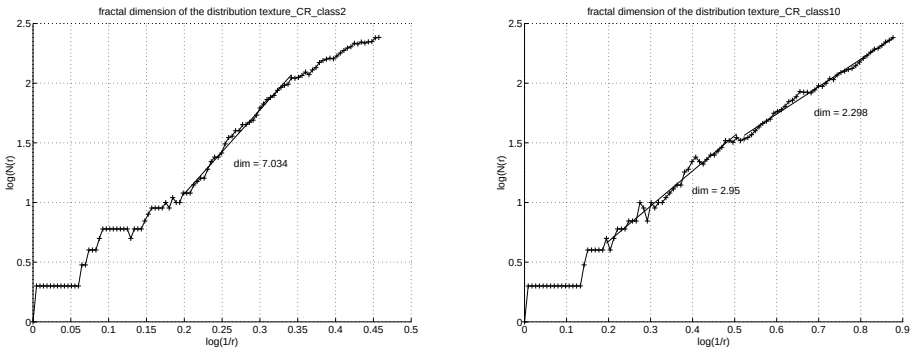


Figure 3.16: Plot of $N(r)$ for class 2 and class 10 of the *Texture* database.

| *Class* | *2* | *3* | *4* | *6* | *7* | *8* | *9* | *10* | *12* | *13* | *14* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *2* | 0 | 1.10 | 2.46 | 3.54 | 4.38 | 1.68 | 0.76 | 1.73 | 2.74 | 3.45 | 1.58 |
| *3* | 0.94 | 0 | 3.13 | 3.03 | 3.69 | 1.60 | 0.79 | 1.61 | 3.17 | 3.32 | 1.22 |
| *4* | 1.77 | 2.66 | 0 | 6.31 | 6.88 | 2.92 | 2.05 | 2.83 | 2.15 | 4.96 | 3.19 |
| *6* | 2.07 | 2.08 | 5.10 | 0 | 2.70 | 0.85 | 1.46 | 0.74 | 6.03 | 1.89 | 0.95 |
| *7* | 2.16 | 2.15 | 4.73 | 2.30 | 0 | 1.28 | 1.50 | 1.14 | 5.72 | 2.61 | 1.49 |
| *8* | 2.20 | 2.49 | 5.33 | 1.93 | 3.41 | 0 | 1.46 | 0.35 | 6.26 | 1.63 | 1.10 |
| *9* | 0.86 | 1.05 | 3.20 | 2.81 | 3.42 | 1.25 | 0 | 1.32 | 3.73 | 2.91 | 1.15 |
| *10* | 2.52 | 2.78 | 5.75 | 1.86 | 3.36 | 0.39 | 1.72 | 0 | 6.87 | 1.37 | 1.32 |
| *12* | 1.54 | 2.10 | 1.68 | 5.82 | 6.50 | 2.68 | 1.87 | 2.65 | 0 | 4.79 | 2.75 |
| *13* | 3.48 | 3.95 | 6.96 | 3.28 | 5.33 | 1.25 | 2.62 | 0.95 | 8.60 | 0 | 2.25 |
| *14* | 1.59 | 1.45 | 4.44 | 1.64 | 3.02 | 0.84 | 1.03 | 0.91 | 4.90 | 2.23 | 0 |

Table 3.7: Dispersion matrix computed on the *Texture* database.

| Eigen value | Value | Inertia percentage | Cumulated inertia |
|---|---|---|---|
| 1 | 30.267500 | 75.67 | 75.67 |
| 2 | 3.6512500 | 9.13 | 84.80 |
| 3 | 2.2937000 | 5.73 | 90.53 |
| 4 | 1.7039700 | 4.26 | 94.79 |
| 5 | 0.6716540 | 1.68 | 96.47 |
| 6 | 0.5015290 | 1.24 | 97.72 |
| 7 | 0.1922830 | 0.48 | 98.20 |
| 8 | 0.1561070 | 0.39 | 98.59 |
| 9 | 0.1099570 | 0.27 | 98.87 |
| 10 | 0.0890891 | 0.22 | 99.09 |
| 11 | 0.0656016 | 0.16 | 99.25 |
| 12 | 0.0489988 | 0.12 | 99.38 |
| 13 | 0.0433819 | 0.11 | 99.49 |
| 14 | 0.0345022 | 0.09 | 99.57 |
| 15 | 0.0299203 | 0.07 | 99.65 |
| 16 | 0.0248857 | 0.06 | 99.71 |
| 17 | 0.0167901 | 0.04 | 99.75 |
| 18 | 0.0161633 | 0.04 | 99.79 |
| 19 | 0.0128898 | 0.03 | 99.82 |
| 20 | 0.0113884 | 0.03 | 99.85 |

Table 3.8: The inertia percentages associated to the 20 first eigenvalues of the *Texture* database.

database, there is a good matching between the indications given by the dispersion matrix (table 3.7) evaluated according to equation 1.2 and the K_NN (LOO) confusion matrix (table 3.6). That means that when the coefficient is small in the dispersion matrix, a confusion can exist between the two classes. We can thus deduce that this database is composed of several well identified clusters (one cluster per class).

### 3.4.8 Principal Component Analysis

A principal component analysis was performed on the *Texture* database. The database resulting from this preprocessing is on the ftp server in the "*/REAL/texture/texture_PCA.dat.Z*" file. Table (3.8) provides the inertia percentages associated to the eigenvalues corresponding to the 20 first principal component axis. 99.85 percent of the total database inertia will remain if the 20 first principal components are kept.

# Bibliography


[1] P. Alinat. Periodic Progress Report 4. Technical report, ROARS Project ESPRIT II- Number 5516, February 1993. Thomson report TS. ASM 93/S/EGS/NC/079.

[2] Y. Cheneval. Packlib, an interactive environment to develop modular software for data processing. In Prieto Mira, Cabestany, editor, *IWANN95-Proceedings of the International Workshop on Artificial Neural Networks*, Malaga, Spain, June 1995. Springer-Verlag Lecture Notes in Computer Sciences.

[3] P. Comon. Classification bayésienne distribuée. *Revue Technique Thomson CSF*, 22(4):543–561, 1990.

[4] P. Comon. Classification supervisée par réseaux multicouches (supervised classification by multilayer networks). *Traitement du Signal*, 8(6):387–407, December 1991.

[5] P. Comon. Supervised classification: a probabilistic approach. In M. Verleysen, editor, *ESANN95-European Symposium on Artificial Neural Networks*, Brussels, Belgium, April 1995. D facto publications.

[6] P. Comon et al. Deliverable R1-A-P - Axis A: Theory. Technical report, Elena-NervesII "Enhanced Learning for Evolutive Neural Architecture", ESPRIT-Basic Research Project Number 6891, June 1993.

[7] B.V. Dasarathy. Nosing around the neighborhood: A new system structure and classification rule for recognition in partially exposed environments. In *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No. 1, 67-71*, 1980.

[8] P.A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach*. Prentice Hall, London, 1982.

[9] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.

[10] A. Feelders and W. Verkooijen. Which method learns most from the data ? Technical report, University of Twente, Deparment of Computer Science, P.O. Box 217, 7500 AE Enschede, The Netherlands, Jan. 1995. Anonymous FTP: /pub/doc/pareto/aistats95.ps.Z on ftp.cs.utwente.nl.

[11] C. Feng, A. Sutherland, S. King, S. Muggleton, and R. Henery. Comparison of machine learning classifiers to statistics and neural networks. In *Artificial Intelligence and Statistics Conf. 93*, 1993.

[12] R.A. Fisher. The use of multiple measurements in taxonomic problems. In *Machine Learning Vol 6-2 March*, 1991.

[13] K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, Inc., 1250 Sixth Avenue, San Diego, CA 92101, 2nd edition, 1990.

[14] G.W. Gates. The reduced nearest neighbor rule. In *IEEE Transactions on Information Theory, May, 431-433.*, 1972.

[15] A. Guérin-Dugué and C. Avilez-Cruz. High order statistics from natural textured images. In *ATHOS Workshop on System Identification and High Order Statistics*, Sophia-Antipolis (France), September 1993.

[16] H.G.E. Hentschel and I. Procaia. The infinite number of generalized dimensions of fractals and strange attractors. In *Physica 8D*, pages 435–444, 1983.

[17] T. Kohonen, G. Barna, and R. Chrisley. Statistical pattern recognition with neural networks: Benchmarking studies. In *IEEE Int. Conf. on Neural Networks*, volume 1, San Diego, CA, 1988. SOS Printing.

[18] B.B. Mandelbrot. *Fractal geometry of nature.* Freeman, San Fransisco, 1982.

[19] D. Michie, D.J. Spiegelhalter, and C.C. Taylor, editors. *Machine learning, Neural and Statistical Classification.* Ellis Horwood Series In Artificial Intelligence, England, 1994.

[20] P. M. Murphy and D.W. Aha. Uci repository of machine learning databases, 1991. Irvine, University of California, Department of Information and Computer Science, Anonymous FTP: /pub/machine-learning-database on ics.uci.edu.

[21] G. Nakhaeizadeh. Project STATLOG . Technical report, ESPRIT IPSS-2 Number 5170 : Comparative Testing and Evaluation of Statistical and Logical Learning Algorithms for Large-Scale Applications in Classification, Prediction and Control, April 1993.

[22] Lutz Prechelt. PROBEN1 — A set of benchmarks and benchmarking rules for neural network training algorithms. Technical Report 21/94, Fakultät für Informatik, Universität Karlsruhe, D-76128 Karlsruhe, Germany, September 1994. Anonymous FTP: /pub/papers/techreports/1994/1994-21.ps.Z on ftp.ira.uka.de.

[23] B.D. Ripley. Flexible non-linear approaches to classification. In V. Cherkassky, J.H. Friedman, and H. Wechsler, editors, *From Statistics to Neural Networks: Theory and Pattern Recognition Applications.* Sringer Verlag, 1993.

[24] C. W. Therrien. *Decision estimation and classifiation.* Willey, 1989.

[25] C Trichot. *Courbes et dimension fractale.* Springer-Verlag, 1993.

[26] Zijian Zheng. A benchmark for classifer learning. Technical Report TR474, Basser Department of Computer Science, University of Sidney, N.S.W Australia 2006, 1993. Anonymous FTP: /pub/tr on ftp.cs.su.oz.au.