Principal component analysis of fuzzy data using autoassociative neural networks

T. Denœux[1] and M. Masson

Université de Technologie de Compiègne

U.M.R CNRS 6599 Heudiasyc

Centre de Recherches de Royallieu

BP 20529 - F-60205 Compiègne cedex - France

August 22, 2003

[1]Corresponding author. Email: `tdenoeux@hds.utc.fr`.

Abstract

This paper describes an extension of principal component analysis (PCA) allowing the extraction of a limited number of relevant features from high dimensional fuzzy data. Our approach exploits the ability of linear autoassociative neural networks to perform information compression in just the same way as PCA, without explicit matrix diagonalization. Fuzzy input values are propagated through the network using fuzzy arithmetics, and the weights are adjusted to minimize a suitable error criterion, the inputs being taken as target outputs. The concept of correlation coefficient is extended to fuzzy numbers, allowing the interpretation of the new features in terms of the original variables. Experiments with artificial and real sensory evaluation data demonstrate the ability of our method to provide concise representations of complex fuzzy data.

Keywords: Fuzzy data analysis, Feature extraction, Neural networks, Pattern recognition.

1 Introduction

Data compression and feature extraction are among the main problems addressed in exploratory data analysis and pattern recognition. Given a collection $\mathbf{x}^1, \ldots, \mathbf{x}^n$ of n d-dimensional real vectors describing n objects according to d attributes, it is often desirable, for data visualization or efficient classification, to compress this information into lower-dimensional data $\mathbf{y}^1, \ldots, \mathbf{y}^n$, while preserving as much as possible of the original information. One of the simplest and most widely used methods for feature extraction is known as *principal component analysis* (PCA). Viewing the n data points as a cloud in $\mathbb{R}^d$, PCA captures the main features of the data set by searching for directions along which the dispersion, or variance, of the cloud is maximal. If q such directions are found (with $q < d$), they define a q-dimensional linear subspace $\mathcal{L}$ of $\mathbb{R}^d$, such that the projections along $\mathcal{L}$ of the $\mathbf{x}^p$ ($p = 1, \ldots, n$), having maximal dispersion, constitute, in some sense, an optimal q-dimensional picture of the original data. PCA is used extensively as a tool for discovering the underlying structure of data sets through two-dimensional displays (allowing, e.g., to find groups of similar observations or correlations between variables), or as a preprocessing step prior to other operations such as clustering, regression, or classifier design (see, e.g., [17]).

In this paper, it is proposed to extend PCA to a wider class of data comprising real numbers, real intervals and, more generally, fuzzy numbers. Fuzzy numbers are defined as fuzzy sets of the real line whose α-cuts are closed intervals [16]. They may be viewed as generalized intervals with possibly ill-defined boundaries (real numbers and real intervals are recovered as special cases). In data analysis, fuzzy numbers may be used to model imprecise observations (derived, e.g., from uncertain measurements or linguistic assessments), as well as distributions of values taken by an attribute during repeated measurements, or related to different entities forming a class of interest. Here are three examples (adapted from [31]) illustrating these different situations.

Example 1 Let x denote the speed of a vehicle at some time, and assume that a human observer reports that "x is high". This information may be modeled by a fuzzy set $\widetilde{x}$, the membership function $\mu_{\widetilde{x}}$ of which is agreed to represent the linguistic label "high". For example, it may be a triangular membership function with modal value 100 and support $[90, 110]$.

Example 2 As before, let x be the speed of a vehicle, but let us now assume that x is measured by k sensors. We thus have an empirical distribution $\xi_1, \ldots, \xi_k$ of k numerical observations, ξ_i denoting the value returned by sensor i. This information may, again, be conveniently represented by a fuzzy set $\widetilde{x}$ with unimodal membership function defined using simple statistics of the distribution, such as the sample average and standard deviation, or other indicators of central tendency and dispersion.

Example 3 A third situation is one in which the objects in the database under study consist of collections of entities. For instance, in a study about different categories of cars, an object of interest may consist of "sports cars". Such an object may be described by attributes such as "maximal speed" denoted by x. The value taken by x for the object "sports car" may be a crisp set (e.g., the interval [200,300]), or a fuzzy set $\widetilde{x}$ (e.g., a triangular fuzzy set with modal value 250 and support [200,300]).

Note that the situation illustrated by example 3 is clearly distinct from the previous ones. In examples 1 and 2, the value taken by variable x is unique and well defined, but it is only partially known. The membership function $\mu_{\widetilde{x}}$ of $\widetilde{x}$ is then seen as a possibility distribution quantifying our partial knowledge of x. In example 3, x is a multi-valued attribute, which is known with complete certainty to be equal to the fuzzy set $\widetilde{x}$ for the object under consideration. Because of the formal equivalence between possibility distributions and membership functions of fuzzy sets [37, 15], both situations are amenable to the same formal analysis.

In the rest of this paper, it will be assumed that we have a collection $\widetilde{\mathbf{x}}^1, \ldots, \widetilde{\mathbf{x}}^n$ of n d-dimensional vectors of fuzzy numbers, and our goal will be to define a "generalized PCA" of these data in order to obtain a synthetic representation as n q-dimensional vectors of fuzzy numbers $\widetilde{\mathbf{y}}^1, \ldots, \widetilde{\mathbf{y}}^n$, with $q < d$, carrying most of the relevant information present in the original data.

Although considerable work has been devoted to the development of fuzzy algorithms for analyzing crisp data (see, e.g., [5]), comparatively less attention has been paid, until recently, to the analysis of fuzzy data. References [18] [36] [4] describe recent developments from an inferential statistics perspective, and a review of linear regression analysis of fuzzy data may be found in [13]. In [19, 31], a general approach to the handling of fuzzy data is proposed, in which fuzzy data are mapped onto a crisp representation space where classical algorithms (e.g., clustering procedures) can be applied. This approach, however, is not oriented towards visualizing high dimensional fuzzy data, which is our main concern here. Multidimensional scaling, a technique to map objects to a multidimensional feature space based on observed dissimilarities between objects, has recently been extended to interval-valued and fuzzy dissimilarities [12, 30].

In the framework of symbolic data analysis (SDA) [7], several extensions of PCA to interval or histogram data have been proposed [10] [27] [32]. These approaches generalize classical principles of PCA to hypercube data by applying the standard analysis to the centers or to the vertices of the hypercubes [7, chapter 9]. The simplicity of both approaches has intuitive appeal. However, the "centers method" does not take into account the imprecision of the data in the feature extraction process and, consequently, builds only suboptimal low-dimensional representations of the data, as will be shown experimentally in Section 4.1. The other approach, dubbed the "vertices method" in [7], is not adapted to large high-dimensional data sets, since the analysis is carried out with $n2^d$ input vectors.

The new approach presented in this paper attempts to overcome these limitations by finding iteratively linear features from which the original fuzzy data may be recovered with minimum error using a linear transformation. The proposed method builds upon two groups of results available in the literature. The first one concerns the implementation of standard PCA for crisp data using a neural network [8, 2, 3]. The architecture is that of a three-layer autoassociative linear neural network composed of an input layer and an output layer of d neurons each, and a hidden layer of q linear units. By minimizing a suitable error criterion, such a network has been shown to develop in its hidden layer insightful representations, the visualization of which reveals the most salient features of the original data. The second research direction that has inspired our approach is related to the fuzzification of neural networks. Many different models have been proposed, including multilayered networks with fuzzy inputs, real weights and fuzzy outputs [22], multilayered networks with crisp inputs, fuzzy weights and fuzzy outputs [20] and multilayered networks with fuzzy inputs, fuzzy weights and fuzzy outputs [23] [24] [28]. In this paper, only the first kind of model will be considered, as explained in Section 3.

The rest of the paper is organized as follows. Section 2 recalls the necessary background about PCA and its implementation in autoassociative neural networks (ANN). Section 3 presents the extension of PCA to fuzzy data based on the "fuzzification" of ANN's. Lastly, experimental results with artificial and real data are presented in Section 4.

2 PCA and autoassociative neural networks

2.1 Principal component analysis

The purpose of this section is to summarize some basic mathematical facts about PCA. More details may be found in standard textbooks on multivariate analysis such as [1].

Let $\mathbf{x}^1, \ldots, \mathbf{x}^n$ be the n d-dimensional real vectors constituting the data set. Without loss of generality, we shall assume the data to be centered, i.e.

$$\frac{1}{n}\sum_{p=1}^{n} \mathbf{x}^p = 0.$$

With this assumption, we can think of the n data points as a cloud in d-dimensional Euclidean space, with center of gravity located at the origin. As mentioned in Section 1, PCA attempts to find a q-dimensional subspace $\mathcal{L}$ of $\mathbb{R}^d$, such that the orthogonal projections $P_{\mathcal{L}}\mathbf{x}^p$ of the n points on $\mathcal{L}$ have maximal variance.

If $\mathcal{L}$ is the line spanned by unit vector $\mathbf{u}$, the projection of $\mathbf{x} \in \mathbb{R}^d$ on $\mathcal{L}$ is

$$P_{\mathcal{L}}\mathbf{x} = (\mathbf{u}'\mathbf{x})\mathbf{u},$$

where prime denotes transposition. The variance of the data in the direction of $\mathcal{L}$ is therefore

$$\begin{aligned}
\frac{1}{n}\sum_{p=1}^{n}(\mathbf{u}'\mathbf{x}^p)^2 &= \frac{1}{n}\sum_{p=1}^{n}\mathbf{u}'\mathbf{x}^p\mathbf{x}^{p\prime}\mathbf{u} \\
&= \mathbf{u}'\left(\frac{1}{n}\sum_{p=1}^{n}\mathbf{x}^p\mathbf{x}^{p\prime}\right)\mathbf{u} \\
&= \mathbf{u}'S\mathbf{u}
\end{aligned}$$

where S is the sample covariance matrix of the data. PCA thus looks for the vector $\mathbf{u}^*$ which maximizes $\mathbf{u}'S\mathbf{u}$, under the constraint $\|\mathbf{u}\| = 1$. It is easy to show that the solution is the normalized eigenvector $\mathbf{u}_1$ of S associated to its largest eigenvalue λ_1, and

$$\mathbf{u}_1'S\mathbf{u}_1 = \lambda_1\mathbf{u}_1'\mathbf{u}_1 = \lambda_1.$$

There is hardly any difficulty in extending this argument to find the q-dimensional subspace $\mathcal{L}$ on which the projected points $P_{\mathcal{L}}\mathbf{x}^p$ have maximal variance. For mathematical simplicity, let us assume that S is of full rank and has no multiple eigenvalues[1], so that its eigenvalues may be noted $\lambda_1 > \lambda_2 > \ldots > \lambda_d$, with corresponding normalized eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_d$. Then the following proposition holds:

Proposition 1

Among all q-dimensional subspaces $\mathcal{L}$ of $\mathbb{R}^d$, the one spanned by the first q normalized eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_q$ of S is such that $\frac{1}{n}\sum_{p=1}^{n}\|P_{\mathcal{L}}\mathbf{x}^p\|^2$ is maximal. Equivalently, it is also the subspace $\mathcal{L}$ such that the average projection error

$$\frac{1}{n}\sum_{p=1}^{n}\|\mathbf{x}^p - P_{\mathcal{L}}\mathbf{x}^p\|^2$$

is minimized.

[1] This condition is almost always observed with real data. Anyway, it could always be enforced by perturbing the data by small amounts.

The lines spanned by the eigenvectors $\mathbf{u}_j$ are called the principal axes of the data, and the q new features $y_j = \mathbf{u}_j'\mathbf{x}$ defined by the coordinates of $\mathbf{x}$ along the principal axes are called the principal components. The vector $\mathbf{y}^p$ of principal components for each initial pattern vector $\mathbf{x}^p$ $(p = 1, \ldots, n)$ may easily be computed in matrix form as $\mathbf{y}^p = U_q'\mathbf{x}^p$, where $U_q = [\mathbf{u}_1, \ldots, \mathbf{u}_q]$ is the $d \times q$ matrix having the q normalized eigenvectors of S as its columns. The variance of the j-th component is

$$\frac{1}{n}\sum_{p=1}^{n} \mathbf{u}_j'\mathbf{x}^p\mathbf{x}^{p\prime}\mathbf{u}_j = \mathbf{u}_j'S\mathbf{u}_j = \lambda_j.$$

2.2 Autoassociative linear networks

As first noticed by Bourlard and Kamp [8], there is an interesting connection between PCA and autoassociative multilayer perceptrons.

Let us consider a feedforward three-layer neural network with an input layer of d units, a hidden layer of size q, and an output layer of d units (Figure 1). Let A be the $q \times d$ matrix of input-to-hidden weights, and let B be the $d \times q$ matrix of hidden-to-output weights. The hidden and output units are assumed to have identity transfer functions, so that the network output $\mathbf{z}$ is computed from the input vector $\mathbf{x}$ as

$$\mathbf{z} = BA\mathbf{x}. \tag{1}$$

Let us assume that this network is trained in autoassociative mode, i.e., using the inputs as target outputs. The network then learns to approximate the identity mapping. If $q < d$, such a task will force the system to find efficient ways of compressing the information contained in the input patterns: the network will work as an unsupervised feature extractor.

The use of such a scheme for information compression and dimensionality reduction was first suggested by Rumelhart et al. [34]. It was analyzed formally by Bourlard and Kamp [8] using the concept of singular value decomposition of matrices. Further results were obtained by Baldi and Hornik [2, 3], who provided a complete description of the error surfaces of multilayer linear networks (of which autoassociative networks with one hidden layer are a special case). A review of applications to the problem of face recognition in images can be found in [35].

The findings of Baldi and Hornik for the autoassociative case may be briefly summarized as follows. Let $E(A, B)$ denote the quadratic error function, defined as:

$$E(A, B) = \sum_{p=1}^{n} e(\mathbf{x}^p, \mathbf{z}^p), \tag{2}$$

where $e(\mathbf{x}^p, \mathbf{z}^p)$ denotes the reconstruction error for pattern p:

$$e(\mathbf{x}^p, \mathbf{z}^p) = \|\mathbf{x}^p - \mathbf{z}^p\|^2 = \sum_{k=1}^{d} (x_k^p - z_k^p)^2, \quad p = 1, \ldots, n. \tag{3}$$

The total error may also be expressed as a function of the global map $W = BA$, which is constrained to be at most of rank q. It is clear that $E(A, B) = E(CA, BC^{-1})$ for any invertible $q \times q$ matrix C, so that any global map W corresponds to an infinite family of weight matrices (A, B).

As before, let S denote the sample covariance matrix of the data, assumed to be of full rank, with eigenvalues $\lambda_1 > \ldots, \lambda_d$ and corresponding orthonormal eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_d$. The following proposition of Baldi and Hornik [2] may be expressed as follows:

Proposition 2

The error E expressed as a function of the global map W has a unique local and global minimum of the form $W = BA$ with

$$A = CU_q' \tag{4}$$

$$B = U_q C^{-1}, \tag{5}$$

where U_q denotes, as before, the matrix $[\mathbf{u}_1, \ldots, \mathbf{u}_q]$, and C is an arbitrary invertible $q \times q$ matrix.

The optimal map $W = U_q U_q'$ is thus the orthogonal projection $P_{\mathcal{L}}$ onto the subspace $\mathcal{L}$ spanned by the first eigenvectors of the data covariance matrix S. PCA is recovered as a special case when C is the identity matrix: in that case, the activities in the hidden layer are exactly identical to the principal components of the data. If the error function is minimized by an iterative algorithm such as backpropagation, this particular solution is, however, generally not obtained, and matrix C is arbitrary. The vector of hidden unit activities is then

$$A\mathbf{x} = CU_q'\mathbf{x} = C\mathbf{y}$$

where $\mathbf{y}$ is the vector of principal components for input $\mathbf{x}$. Therefore, the hidden unit activities are identical to the principal components, *up to an arbitrary linear transformation*. Although such a general solution may have some advantages in terms of robustness from the information compression viewpoint [3], the internal representations produced are not directly usable in data analysis because the scaling of the hidden layer activities is completely arbitrary. A way to resolve this ambiguity is to introduce the constraint $A' = B$, which given Eqs (4) and (5) translates to

$$U_q C' = U_q C^{-1}. \tag{6}$$

Since the $(\mathbf{u}_1, \ldots, \mathbf{u}_q)$ form an orthonormal basis, we have $U_q' U_q = I_q$, where I_q is the identity matrix of size q. By left-multiplying both sides of Eq. (6) by U_q', we see that C verifies $C' = C^{-1}$ or, equivalently, $C'C = I_d$. Hence, C is now an orthogonal matrix, which implies that the hidden unit activities and the principal components are related by an isometric transformation (the group of isometric transformations includes rotations and reflections). A simple way to impose the constraint $A' = B$ is to re-write the propagation equation (1) as

$$\mathbf{z} = BB'\mathbf{x} \tag{7}$$

which translates in scalar notation to

$$z_k = \sum_{j=1}^{q} B_{kj} \sum_{i=1}^{d} B_{ij} x_i \quad k = 1, \ldots, d. \tag{8}$$

3 Extension to fuzzy data

3.1 Principle

Let us now assume that we have a collection of n objects described by d attributes taking values in the set $\mathcal{F}(\mathbb{R})$ of real fuzzy numbers. The data thus takes the form

$\widetilde{\mathbf{x}}^1, \ldots, \widetilde{\mathbf{x}}^n$, where each $\widetilde{\mathbf{x}}^p \in \mathcal{F}(\mathbb{R})^d$ is a vector of d fuzzy numbers noted $(\widetilde{x}_i^p)_{1 \leq i \leq d}$.

We want to compress this data into lower dimensional fuzzy data $\widetilde{\mathbf{y}}^1, \ldots, \widetilde{\mathbf{y}}^n$, with $\widetilde{\mathbf{y}}^p \in \mathcal{F}(\mathbb{R})^q$, $p = 1, \ldots, n$, and $q < d$. In the case of real data, this problem is usually solved by PCA. How can this approach be generalized to the case of fuzzy data ?

A possible answer to this question may be found using the neural network implementation of PCA described in the previous section. Let us consider again the three layer network depicted in Figure 1, with linear hidden and output units, and assume that a vector $\widetilde{\mathbf{x}}$ of d fuzzy numbers is fed into the input layer. The network output may be computed by applying Zadeh's extension principle to Eq. (8). The k-th component $\widetilde{z}_k$ of the fuzzy output vector $\widetilde{\mathbf{z}}$ for the fuzzy input $\widetilde{\mathbf{x}}$ is then defined as

$$\forall u \in \mathbb{R} \quad \mu_{\widetilde{z}_k}(u) = \sup_{v_1, \ldots, v_d} \min_{1 \leq i \leq d} \mu_{\widetilde{x}_i}(v_i), \tag{9}$$

the supremum being taken under the constraint

$$u = \sum_{j=1}^{q} B_{kj} \sum_{i=1}^{d} B_{ij} v_i.$$

We may write in more compact form

$$\widetilde{z}_k = \sum_{j=1}^{q} B_{kj} \sum_{i=1}^{d} B_{ij} \widetilde{x}_i \quad k = 1, \ldots, d, \tag{10}$$

where addition and multiplication by a real are now the usual operations of fuzzy arithmetics [16]. The practical calculation of the $\widetilde{z}_k$ in the special case of trapezoidal fuzzy numbers will be addressed in the next section. Using matrix notation, Eq. (10) translates naturally to

$$\widetilde{\mathbf{z}} = BB'\widetilde{\mathbf{x}},$$

which is the fuzzy counterpart of Eq. (7).

Let us denote by $\widetilde{\mathbf{y}}$ the vector of fuzzy hidden unit activities, defined as $\widetilde{\mathbf{y}} = B'\widetilde{\mathbf{x}}$. This vector constitutes an internal representation of input pattern $\widetilde{\mathbf{x}}$. The internal representations of the n training vectors $\widetilde{\mathbf{x}}^p$ may be globally optimized by generalizing the quadratic error criterion given by Eqs (2-3) to fuzzy outputs and fuzzy target values. This may be achieved by defining some measure of discrepancy or dissimilarity between membership functions (see, e.g., Refs. [21] and [14, page 496] for developments on this topic). Let $e(\tilde{x}, \tilde{z})$ denote such a dissimilarity measure between real fuzzy numbers $\tilde{x}$ and $\tilde{z}$ (an explicit expression of $e(\tilde{x}, \tilde{z})$ in the special case of trapezoidal fuzzy numbers is given in the next section). Then, the reconstruction error for pattern p can be defined as:

$$e(\widetilde{\mathbf{x}}, \widetilde{\mathbf{z}}) = \sum_{k=1}^{d} e(\widetilde{x}_k, \widetilde{z}_k), \tag{11}$$

and the total error over the training set becomes:

$$E(B) = \sum_{p=1}^{n} e(\widetilde{\mathbf{x}}^p, \widetilde{\mathbf{z}}^p) = \sum_{p=1}^{n} \sum_{k=1}^{d} e(\widetilde{x}_k^p, \widetilde{z}_k^p). \tag{12}$$

Adopting a suitable parameterization of the fuzzy numbers (see Section 3.2 for details on the practical implementation), it appears that $E(B)$ is nonlinear with respect to the elements of B. Let B^* be a solution of the non-linear optimization problem

$$\min_B E(B).$$

This problem can be solved using a standard iterative gradient-based procedure. The hidden layer output vectors $\widetilde{\mathbf{y}}^p = B^* \widetilde{\mathbf{x}}^p$ constitute "optimal" q-dimensional representations of the original input vectors $\widetilde{\mathbf{x}}^p$, in the sense that they allow the original vectors to be recovered with minimal error by a linear transformation.

A geometric view of the operations performed by the autoassociative network described above is provided by Figure 2, for the special case of two-dimensional inputs ($d = 2$) and one hidden unit ($q = 1$). In that case, the weight matrix B is of size 2×1 and may be seen as vector of $\mathbb{R}^2$. Let us consider the operations performed for a fuzzy input vector $\widetilde{\mathbf{x}} = (\widetilde{x}_1, \widetilde{x}_2)'$ with trapezoidal fuzzy components. The hidden unit computes the scalar product between B and $\widetilde{\mathbf{x}}$, which amounts to projecting $\widetilde{\mathbf{x}}$ onto the line $\mathcal{L}$ spanned by B. The hidden unit activity is a fuzzy number $\widetilde{y}$, whose α-cut for any $\alpha \in]0, 1]$ is the projection on $\mathcal{L}$ of the set $(\widetilde{x}_1)_\alpha \times (\widetilde{x}_2)_\alpha$, where $(\widetilde{x}_1)_\alpha$ and $(\widetilde{x}_2)_\alpha$ denote, respectively, the α-cut of $\widetilde{x}_1$ and $\widetilde{x}_2$. The computation of the network outputs then amounts to projecting back $\widetilde{y}$ onto the two initial axes x_1 and x_2, yielding a vector $\widetilde{\mathbf{z}} = (\widetilde{z}_1, \widetilde{z}_2)'$ of reconstructed outputs. The learning process consists in the determination of the weight vector B minimizing a measure of discrepancy $E(B)$ between the original patterns $\widetilde{\mathbf{x}}^p$ and the reconstructed vectors $\widetilde{\mathbf{z}}^p$.

Remark 1 It could be tempting to further generalize the autoassociative network described in Section 2.2 by allowing the weights to be fuzzy. Such fuzzy neural networks have been proposed by various authors in a classification or regression context (see, e.g., [9, 23, 24]). However, for such networks, the hidden unit activities can no longer be understood in terms of projection on linear subspaces, and the interpretation of the representations produced seems extremely difficult, which reduces the interest of such an approach for exploratory data analysis. For the same reason, the inclusion of additional hidden layers and non-linearities, which is, of course, possible and useful for data compression purposes, has not been considered in this study.

Remark 2 An alternative, and perhaps simpler approach to extending PCA to fuzzy data could be to (1) defuzzify the data (e.g., by reducing each fuzzy number to its expected value [21]), (2) perform standard PCA on the real data, and (3) project the fuzzy data onto the subspace spanned by the first q eigenvectors obtained in the previous step. This is, indeed, the idea behind the "Centers method" proposed by Cazes et al. [10] in the case of interval-valued data. This approach, however, will be shown in Section 4.1 to overestimate the ambiguity in the input data by producing overly imprecise representations.

Remark 3 In the expression of the error function given by Eq. (12), each error term $e(\widetilde{x}_k^p, \widetilde{z}_k^p)$ is given the same weight. This convention may be questioned in the case of highly heterogeneous data, some inputs $\widetilde{x}_k^p$ having much more imprecision than others. In such a case, it might be reasonable to decrease the relative influence of the most imprecise data items. This may be easily achieved as follows. Let $\widetilde{x}$ be a fuzzy number whose α-cuts are closed intervals denoted by $[(\widetilde{x})_\alpha^-; (\widetilde{x})_\alpha^+]$. Following Delgado [11], let us define the ambiguity of $\widetilde{x}$ as

$$\mathrm{Amb}(\widetilde{x}) = \int_0^1 [(\widetilde{x})_\alpha^+ - (\widetilde{x})_\alpha^-] \, d\alpha,$$

and let $\varphi : \mathbb{R} \mapsto [0,1]$ be a decreasing function. Then the error function $E(B)$ in Eq. (12) may be modified as:

$$E_a(B) = \sum_{p=1}^{n} \sum_{k=1}^{d} \varphi[\mathrm{Amb}(\widetilde{x}_k^p)] e(\widetilde{x}_k^p, \widetilde{z}_k^p). \tag{13}$$

Note that other measures of the imprecision of a fuzzy number, such as nonspecificity [26, page 67] could also be used.

3.2 Practical implementation

Let us now assume the components $\widetilde{x}_i$ of each input vector $\widetilde{\mathbf{x}} \in \mathcal{F}(\mathbb{R})^d$ to be trapezoidal fuzzy numbers $\widetilde{x}_i = (x_i^{(1)}, x_i^{(2)}, x_i^{(3)}, x_i^{(4)})$ (see Appendix A for a reminder on trapezoidal fuzzy numbers). This class of fuzzy numbers is known to be closed under the operations of addition, subtraction, and multiplication by a real number. Consequently, the outputs $\widetilde{z}_k$ of the network are also trapezoidal fuzzy numbers. The calculation of the network outputs and the error function in that special case is detailed in the sequel.

Let us first compute the output $\widetilde{y}_j$ of the j-th hidden unit. By definition

$$\widetilde{y}_j = \sum_{i=1}^{d} B_{ij} \widetilde{x}_i = (y_j^{(1)}, y_j^{(2)}, y_j^{(3)}, y_j^{(4)}) \quad j = 1, \ldots, q. \tag{14}$$

Using the properties described by Eqs (34-36) in Appendix A, we have

$$y_j^{(1)} = \sum_{\substack{i=1 \\ B_{ij}>0}}^{d} B_{ij} x_i^{(1)} + \sum_{\substack{i=1 \\ B_{ij}<0}}^{d} B_{ij} x_i^{(4)}, \tag{15}$$

$$y_j^{(2)} = \sum_{\substack{i=1 \\ B_{ij}>0}}^{d} B_{ij} x_i^{(2)} + \sum_{\substack{i=1 \\ B_{ij}<0}}^{d} B_{ij} x_i^{(3)}, \tag{16}$$

$$y_j^{(3)} = \sum_{\substack{i=1 \\ B_{ij}>0}}^{d} B_{ij} x_i^{(3)} + \sum_{\substack{i=1 \\ B_{ij}<0}}^{d} B_{ij} x_i^{(2)}, \tag{17}$$

$$y_j^{(4)} = \sum_{\substack{i=1 \\ B_{ij}>0}}^{d} B_{ij} x_i^{(4)} + \sum_{\substack{i=1 \\ B_{ij}<0}}^{d} B_{ij} x_i^{(1)}. \tag{18}$$

Similarly, the k-th output is, by definition:

$$\widetilde{z}_k = \sum_{j=1}^{q} B_{kj} \widetilde{y}_j = (z_k^{(1)}, z_k^{(2)}, z_k^{(3)}, z_k^{(4)}) \quad k = 1, \ldots, d, \tag{19}$$

and we have

$$z_k^{(1)} = \sum_{\substack{j=1 \\ B_{kj}>0}}^{q} B_{kj} y_j^{(1)} + \sum_{\substack{j=1 \\ B_{kj}<0}}^{q} B_{kj} y_j^{(4)}, \tag{20}$$

$$z_k^{(2)} = \sum_{\substack{j=1 \\ B_{kj}>0}}^{q} B_{kj} y_j^{(2)} + \sum_{\substack{j=1 \\ B_{kj}<0}}^{q} B_{kj} y_j^{(3)}, \tag{21}$$

$$z_k^{(3)} = \sum_{\substack{j=1 \\ B_{kj}>0}}^{q} B_{kj} y_j^{(3)} + \sum_{\substack{j=1 \\ B_{kj}<0}}^{q} B_{kj} y_j^{(2)}, \tag{22}$$

$$z_k^{(4)} = \sum_{\substack{j=1 \\ B_{kj}>0}}^{q} B_{kj} y_j^{(4)} + \sum_{\substack{j=1 \\ B_{kj}<0}}^{q} B_{kj} y_j^{(1)}. \tag{23}$$

The above equations can be also formulated using simple matrix algebra. Let B^+ and B^- denote the positive and negative parts of matrix B:

$$B_{ij}^+ = \begin{cases} B_{ij} & \text{if } B_{ij} > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{24}$$

$$B_{ij}^- = \begin{cases} B_{ij} & \text{if } B_{ij} < 0 \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

Let $\mathbf{x}^{(\ell)} = (x_1^{(\ell)}, ..., x_d^{(\ell)})'$ denote the d-dimensional vector of the ℓ-th parameters of the fuzzy numbers in the input layer. In the same way, let $\mathbf{y}^{(\ell)} = (y_1^{(\ell)}, ..., y_d^{(\ell)})'$ and $\mathbf{z}^{(\ell)} = (z_1^{(\ell)}, ..., z_d^{(\ell)})'$ denote the d-dimensional vectors of the ℓ-th parameters of the fuzzy numbers in the hidden layer and the ouput layer. Then, Eqs (15)-(18) can be written in more compact form as:

$$\mathbf{y}^{(\ell)} = B^{+'}\mathbf{x}^{(\ell)} + B^{-'}\mathbf{x}^{(4-\ell+1)}, \quad \ell \in \{1,2,3,4\}. \tag{26}$$

Similarly, Eqs (20)-(23) can be written as:

$$\mathbf{z}^{(\ell)} = B^{+}\mathbf{y}^{(\ell)} + B^{-}\mathbf{y}^{(4-\ell+1)}, \quad \ell \in \{1,2,3,4\}. \tag{27}$$

The analytical expression of the error function in (12) can be derived by choosing a suitable metric. Considering each trapezoidal fuzzy number as a point in a four-dimensional Euclidean space as proposed in [21] and [13], we define the reconstruction error of the k-th component $\widetilde{x}_k$ by:

$$e(\widetilde{x}_k, \widetilde{z}_k) = \sum_{\ell=1}^{4} (z_k^{(\ell)} - x_k^{(\ell)})^2, \quad k = 1, \ldots, d, \tag{28}$$

and

$$E(B) = \sum_{p=1}^{n} \sum_{k=1}^{d} e(\widetilde{x}_k^p, \widetilde{z}_k^p).$$

The minimization of E with respect to the weights is done using a gradient descent procedure. The algorithm for evaluating the derivatives of E with respect to B is similar to standard backpropagation [34] and is described in Appendix B.

Remark 4 Concerning the complexity of the method, Eqs. (15) to (23) and details given in Appendix B show that both one propagation through the network and one iteration of the gradient calculation (backpropagation) can be performed in $O(ndq)$ operations. This is much less than the standard way of computing PCA (which costs $O(nd^2)$ operations for the computation of the covariance matrix, and $O(d^3)$ operations for diagonalization [6]). Hence, our method seems suitable for processing very large data sets (alternative efficient procedures are based, e.g. on the EM algorithm [33], but they have not been extended to fuzzy data). As an indication, for the sensory data set described in Section 4.2, which involves 252 observations in a five-dimensional space, a complete run of our algorithm implemented in Matlab takes about 5 seconds on a PC equipped with a Pentium II processor.

3.3 Correlation between principal components and initial variables

The interpretation of principal components in terms of the original variables is a fundamental step in the application of PCA to exploratory data analysis. This is usually achieved by computing the linear correlation coefficients between the initial variables x_i and the principal components y_j [25, page 14]. Each principal component is then interpreted as an aggregation of those original variables with which it has strong positive or negative correlation. In our case, both the original variables $\widetilde{x}_i$ and the principal components $\widetilde{y}_j$ are fuzzy, and the notion of correlation coefficient needs to be extended in this context. A *fuzzy* correlation coefficient, whose computation is based on Zadeh's extension principle, has been recently proposed [29]. Its principle is given below. Let $(x^p, y^p), p = 1, n$ denote n pairs of crisp observations. The classical crisp correlation coefficient between the two series of observations x and y is defined as

$$r_{xy} = \frac{\sum_{p=1}^{n}(x^p - \bar{x})(y^p - \bar{y})}{\sqrt{\sum_{p=1}^{n}(x^p - \bar{x})^2 \sum_{p=1}^{n}(y^p - \bar{y})^2}}. \tag{29}$$

Let us now assume that observations consist of n pairs $(\tilde{x}^p, \tilde{y}^p)$ of fuzzy numbers, characterized by their respective membership functions $\mu_{\tilde{x}^p}$ and $\mu_{\tilde{y}^p}$. Let $\tilde{r}$ denote the fuzzy correlation between $\tilde{x}$ and $\tilde{y}$. The extension principle states that

$$\forall r \in \mathbb{R} \quad \mu_{\tilde{r}}(r) = \sup_{\{x^1,\ldots,x^n,y^1,\ldots,y^n / r = r_{xy}\}} \min_{p} \left(\mu_{\tilde{x}^p}(x^p) \wedge \mu_{\tilde{y}^p}(y^p)\right), \tag{30}$$

where $\mu_{\tilde{r}}$ denotes the membership function of $\tilde{r}$, and $\wedge$ denotes the minimum operator. More precisely, let $[(\tilde{x}^p)^-_\alpha; (\tilde{x}^p)^+_\alpha]$ and $[(\tilde{y}^p)^-_\alpha; (\tilde{y}^p)^+_\alpha]$ denote the closed intervals resulting from an α-cut of $\tilde{x}^p$ and $\tilde{y}^p$. Then, each α-cut of $\tilde{r}$ is a closed interval $[(\tilde{r})^-_\alpha; (\tilde{r})^+_\alpha]$ whose bounds are respectively found by solving the following pair of non-linear programs:

$$\begin{aligned}
(\tilde{r})^-_\alpha \quad &= \quad \min_{x^1,\ldots,x^n,y^1,\ldots,y^n} \frac{\sum_{p=1}^{n}(x^p - \bar{x})(y^p - \bar{y})}{\sqrt{\sum_{p=1}^{n}(x^p - \bar{x})^2 \sum_{p=1}^{n}(y^p - \bar{y})^2}} \\
&\text{s.t.} \quad (\tilde{x}^p)^-_\alpha \leq x^p \leq (\tilde{x}^p)^+_\alpha \quad \forall p, \\
&\qquad\;\; (\tilde{y}^p)^-_\alpha \leq y^p \leq (\tilde{y}^p)^+_\alpha \quad \forall p,
\end{aligned} \tag{31}$$

$$\begin{aligned}
(\tilde{r})^+_\alpha \quad &= \quad \max_{x^1,\ldots,x^n,y^1,\ldots,y^n} \frac{\sum_{p=1}^{n}(x^p - \bar{x})(y^p - \bar{y})}{\sqrt{\sum_{p=1}^{n}(x^p - \bar{x})^2 \sum_{p=1}^{n}(y^p - \bar{y})^2}} \\
&\text{s.t.} \quad (\tilde{x}^p)^-_\alpha \leq x^p \leq (\tilde{x}^p)^+_\alpha \quad \forall p, \\
&\qquad\;\; (\tilde{y}^p)^-_\alpha \leq y^p \leq (\tilde{y}^p)^+_\alpha \quad \forall p.
\end{aligned} \tag{32}$$

For a given α-cut, the resolution of (31) and (32) can be done using standard non-linear programming solvers. Note that, due to the nonlinear relationship between the observations and the correlation coefficient, even if $\tilde{x}$ and $\tilde{y}$ are trapezoidal fuzzy numbers, the fuzzy correlation coefficient is not a trapezoidal fuzzy number. In practice, the two non-linear programs are solved for a small finite number of α-cuts, providing a reasonable approximation of $\tilde{r}$.

4 Results

4.1 Simulated data

To illustrate the ability of the proposed method to provide a condensed view of multi-dimensional data, let us consider the hypothetical data set shown in Table 1, adapted from [16, page 237]. The data consist in marks obtained by 6 students in mathematics (M1 and M2) and physics (P1 and P2) during two consecutive terms. Some of these marks are not precisely known and are represented by intervals or linguistic labels, resulting in highly heterogeneous data. By associating a membership function to each linguistic label (see Figure 3), each mark may be represented as a trapezoidal fuzzy number.

A two-dimensional representation of the data was generated by a two-hidden-unit autoassociative network ($q = 2$). The weights of the network are shown in Table 2. They allow some interpretation of the new features: the first axis is linked positively to the attainments in mathematics and the second axis is linked essentially to P1. This is confirmed by the fuzzy correlation coefficients shown in Figure 5. The first axis is clearly correlated with M1 and M2. For the second axis, the correlations are less clear, because of the large imprecision of Tom's attainment in P1. Figure 4 shows a plot of the data along the two axes. Several important characteristics of the original data are recovered in this reduced representation space. For example, Jack, with crisp and rather bad marks, is represented as a crisp object in the lower part of the figure. The wide spread of Tom along the second dimension is explained by the fact that one of its mark in physics is unknown. Bob is very precisely situated on the mathematics axis and less precisely on the other.

A display of the input and reconstructed data is shown in Figure 6. Such a display reveals which aspects of the original data are well preserved in the reduced representation space, and which are not: for instance, Bob's P1 value is well reconstructed, whereas Jack's P2 value is not. The total reconstruction errors $e(\widetilde{\mathbf{x}}^p, \widetilde{\mathbf{z}}^p)$ for the six students are reported in Table 3. It may be seen that Tom has the highest reconstruction error, which means that it may not be well represented in the two-dimensional display of Figure 4.

For comparison purpose, the projection of the fuzzy data on the two first axis of a standard PCA performed on the defuzzified data (centroids), is shown in Figure 7. Although some aspects of the initial information are also well recovered, one can see that the representation is much more confused and that the discrimination between the students is not as good as the one obtained with the autoassociative network.

The last experiment which is reported here aims at illustrating the advantages, in some cases, of the modified error function E_a given in Eq. (13). Let us assume that the marks of an additional student, Emma, are available in the following form: M1=10, M2="unknown", P1=10, P2="unknown". As Emma seems to be an average student and only partial information about her attainments is available, her inclusion in the database can reasonably be required to have little influence on the representation. Two representations were generated using both error functions E and E_a defined, respectively, by Eqs (12) and (13). The weights between the principal axes and the initial variables are shown in Table 4. As expected, the use of error function E_a prevents the determination of the principal axes from being too much influenced by Emma's highly imprecise marks in P2 and M2. The projections of the students are given in Figure 8, in which a greater stability of the second representation can be observed. Criterion E_a therefore appears to be a good alternative to the standard error function E when some items in the database have highly imprecise features and are likely to have an excessive influence on the constructed representation.

4.2 Sensory evaluation data

The proposed method was applied to sensory evaluation data as part of a research project performed in collaboration with a French car manufacturer. The entities under study were noises recorded inside several vehicles. The data consisted in scores given by 12 judges describing their perception of 21 sounds according to 5 attributes. Each sound was presented three times to each subject, yielding a four-way data matrix:

sounds $\times$ attributes $\times$ subjects $\times$ replications. The aim of this work was to study the variability of the responses among the panelists and the variability of each subject during repetitions.

To this end, each of the 21×12 pairs (sound, subject) was considered as an object described by five fuzzy attributes. For each attribute, the three scores available from replications were converted into a triangular fuzzy number (which is a special case of trapezoidal fuzzy number with $x^{(2)} = x^{(3)}$) defined by the minimum, maximum and median value. We thus obtained a set of 12×21 vectors composed of 5 triangular fuzzy numbers. An autoassociative network with two hidden units ($q = 2$) was used to visualize the data. As the imprecision was quite homogeneous in the data, the standard error function E was chosen for the optimization process. The weights obtained are presented in Table 5, and the fuzzy correlation coefficients are shown in Figure 11. They show that the two axes are essentially linked to the second and fourth attributes, respectively.

A first part of the results is shown in Figure 9. For clarity, the responses of the twelve subjects are represented on separate figures for four different sounds. These figures exhibit four typical behaviors of the panelist group. Sound 1 is perceived in a similar way by all the panelists with a very low variability among replications. Judgments for sound 5 are also in good agreement, with a somewhat larger variability in replications. Sound 16 exhibits the same behavior, except for one judge who clearly disagrees with the majority of the group. Sound 21 seems to be difficult to score with a high variability of the responses among panelists and during the replications. Note that the simultaneous representation of the cores and 0.5-cuts of the projections provides useful information regarding the distribution of the assessor responses during replications: for example, the representation will be very different for a judge having answered (0,0,10) for a given sound and a given attribute and another one having answered (0,5,10).

Figure 10 shows a display of the responses given by two assessors. Two extreme cases of panelist behavior are presented. Assessor 4 seems to score the different sounds with a low amplitude in his marks, and with a good repeatability. On the contrary, assessor 12 uses a larger range of marks but with a higher variability among replications.

By carefully studying the different representations, it is thus possible to answer some questions that typically arise in sensory analysis such as: Are the products well discriminated by the attributes ? For which products is there an overall agreement (or disagreement) among panelists ? Which assessors can be considered as reliable or not ? etc.

5 Conclusions

Fuzzy data naturally arise in a variety of situations in which the uncertainty or imprecision of observations cannot be ignored. For instance, in the sensory evaluation application described in the previous section, the spread of the responses given by each panelist is as important as their central value, and new exploratory data analysis techniques need to be developed to take into account this additional complexity in observations. The technique presented in this paper is an extension of principal component analysis allowing the extraction of a limited number of relevant features from fuzzy data. This method exploits recent results regarding the ability of linear autoassociative neural networks to perform information compression in just the same way as PCA, without explicit matrix diagonalization. Experiments with artificial and real data have demonstrated the ability of our method to provide concise descriptions of complex fuzzy data, reflecting not only their central tendency, but also their imprecision. This work is only a first step towards a more systematic application of neural network and fuzzy logic techniques to the analysis of complex data.

A Parametric representation of fuzzy data and interval arithmetics

A fuzzy number is defined as a normal fuzzy subset $\widetilde{x}$ of $\mathbb{R}$ with compact support, and whose α-cuts are closed intervals [16]. Dubois and Prade make a distinction between *fuzzy intervals* and *fuzzy numbers* depending on the multiplicity or uniqueness of modal values. We use the term "fuzzy number" in its most general sense in this paper. A fuzzy number may be viewed as an elastic constraint acting on a certain variable which is only known to lie "around" a certain value. It generalizes both concepts of real number and closed interval. For the sake of computational efficiency and ease of data manipulation, a special class of fuzzy numbers, called trapezoidal fuzzy numbers, is one of the most commonly used. These numbers are defined as follows:

Definition 1

Let $x^{(1)}, x^{(2)}, x^{(3)}$ and $x^{(4)}$ be four real numbers such that $x^{(1)} < x^{(2)} \leq x^{(3)} < x^{(4)}$. The fuzzy quantity $\widetilde{x}$ defined by the following membership function:

$$\mu_{\widetilde{x}}(u) = \begin{cases} 0 & \text{if } u \leq x^{(1)} \text{ or } u \geq x^{(4)} \\ \dfrac{u - x^{(1)}}{x^{(2)} - x^{(1)}} & \text{if } x^{(1)} \leq u \leq x^{(2)} \\ 1 & \text{if } x^{(2)} \leq u \leq x^{(3)} \\ \dfrac{x^{(4)} - u}{x^{(4)} - x^{(3)}} & \text{if } x^{(3)} \leq u \leq x^{(4)} \end{cases} \tag{33}$$

is called a trapezoidal fuzzy number and is denoted by $\widetilde{x} = (x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)})$.

Trapezoidal fuzzy numbers belong to a wider family of fuzzy numbers called LL-fuzzy numbers [16] which will not be described here.

An attractive characteristic of the class of trapezoidal fuzzy numbers is that it is closed with respect to the operations of addition, subtraction, and multiplication by a real (this is also true for LL-numbers). More precisely, let $\widetilde{x} = (x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)})$ and $\widetilde{y} = (y^{(1)}, y^{(2)}, y^{(3)}, y^{(4)})$ be two trapezoidal fuzzy numbers. Then, it can be shown [16] that:

$$\widetilde{x} + \widetilde{y} = (x^{(1)} + y^{(1)}, x^{(2)} + y^{(2)}, x^{(3)} + y^{(3)}, x^{(4)} + y^{(4)}) \tag{34}$$

$$\widetilde{x} - \widetilde{y} = (x^{(1)} - y^{(4)}, x^{(2)} - y^{(3)}, x^{(3)} - y^{(2)}, x^{(4)} - y^{(1)}) \tag{35}$$

$$\forall a \in \mathbb{R}, \quad a\widetilde{x} = \begin{cases} (ax^{(1)}, ax^{(2)}, ax^{(3)}, ax^{(4)}) & \text{if } a \geq 0 \\ (ax^{(4)}, ax^{(3)}, ax^{(2)}, ax^{(1)}) & \text{if } a < 0 \end{cases} \tag{36}$$

By convention, a real number x will be represented by (x, x, x, x) and a closed interval $[a, b]$ by (a, a, b, b), which allows manipulation of both imprecise and crisp data in the same framework.

B Evaluation of error function derivatives

Popularized by Rumelhart, Hinton and Williams [34], the backpropagation algorithm is an efficient way to compute the derivatives of the error function of a multilayered network with respect to the weights. We give in this appendix the details of the computations involved when applied to fuzzy PCA. Let E denote the error function defined as:

$$E = \sum_{p=1}^{n} e(\widetilde{\mathbf{x}}^p, \widetilde{\mathbf{z}}^p).$$

First, we note that the derivatives of E are simply the sum of the derivatives over all input vectors in the data set. To keep clear notations, we omit the superscript p from the input, hidden and output variables. Each input $\widetilde{\mathbf{x}}$ (resp. output $\widetilde{\mathbf{z}}$) is a d-dimensional vector of trapezoidal fuzzy numbers $\widetilde{x}_i = (x_i^{(1)}, x_i^{(2)}, x_i^{(3)}, x_i^{(4)})$ (resp. $\widetilde{z}_i = (z_k^{(1)}, z_k^{(2)}, z_k^{(3)}, z_k^{(4)})$). We concentrate on the derivatives of the error $e(\widetilde{\mathbf{x}}, \widetilde{\mathbf{z}})$, defined as:

$$e(\widetilde{\mathbf{x}}, \widetilde{\mathbf{z}}) = \sum_{\ell=1}^{4} \sum_{k=1}^{d} (z_k^{(\ell)} - x_k^{(\ell)})^2 = \sum_{\ell=1}^{4} e^{(\ell)},$$

with

$$e^{(\ell)} = \sum_{k=1}^{d} (z_k^{(\ell)} - x_k^{(\ell)})^2.$$

By construction, the first layer of weights in our model is equal to the second layer, so that a weight appears twice in the network: in the first layer for connecting the input unit i to the hidden unit j and in the second layer for connecting the hidden unit j to the output unit i. A small modification of standard backpropagation is needed in order to take into account the fact that same weights are common to several connections. A classical way to deal with these so-called *shared weights*, is to compute individually the derivatives for each weight in the two layers of the network, and then to sum up the two derivatives [34, p. 355]. To compute independently the gradient in the first and in the second layer, let us explicitely introduce two notations, $\dot{B}$ and $\ddot{B}$, according to whether one considers the weights from the input layer to the hidden layer ($\dot{B}$) and the weights from the hidden layer to the output layer ($\ddot{B}$), keeping in mind that they both refer to the same matrix B.

First, let us recall, with these notations, the equations of propagation in the network:

$$y_j^{(\ell)} = \sum_{i=1}^{d} \dot{B}_{ij}^{+} x_i^{(\ell)} + \sum_{i=1}^{d} \dot{B}_{ij}^{-} x_i^{(4-\ell+1)} \quad j = 1, \ldots, q, \quad \ell = 1, \ldots, 4. \tag{37}$$

$$z_k^{(\ell)} = \sum_{j=1}^{q} \ddot{B}_{kj}^{+} y_j^{(\ell)} + \sum_{i=1}^{q} \ddot{B}_{kj}^{-} y_j^{(4-\ell+1)} \quad k = 1, \ldots, d, \quad \ell = 1, \ldots, 4. \tag{38}$$

Gradients in the output layer

$$\frac{\partial e(\widetilde{\mathbf{x}}, \widetilde{\mathbf{z}})}{\partial \ddot{B}_{kj}} = \sum_{\ell=1}^{4} \frac{\partial e^{(\ell)}}{\partial \ddot{B}_{kj}} \quad k = 1, \ldots, d, \quad j = 1, \ldots, q \tag{39}$$

$$\frac{\partial e^{(\ell)}}{\partial \ddot{B}_{kj}} = \frac{\partial e^{(\ell)}}{\partial z_k^{(\ell)}} \frac{\partial z_k^{(\ell)}}{\partial \ddot{B}_{kj}} \quad k = 1, \ldots, d, \quad j = 1, \ldots, q, \quad \ell = 1, \ldots, 4. \tag{40}$$

Let

$$\delta_k^{(\ell)} \triangleq \frac{\partial e^{(\ell)}}{\partial z_k^{(\ell)}} = 2(z_k^{(\ell)} - x_k^{(\ell)}) \quad k = 1, \ldots, d, \quad \ell = 1, \ldots, 4. \tag{41}$$

Using (41) and (38), one obtains the expressions of the partial derivatives with respect to the second-layer weights in the following form:

$$\frac{\partial e^{(\ell)}}{\partial \ddot{B}_{kj}} = \delta_k^{(\ell)} \left(H(\ddot{B}_{kj}) y_j^{(\ell)} + [1 - H(\ddot{B}_{kj})] y_j^{(4-\ell+1)} \right)$$
$$k = 1, \ldots, d, \quad j = 1, \ldots, q, \quad \ell = 1, \ldots, 4, \tag{42}$$

where H is the Heaviside function defined as $H(u) = 1$ if $u \geq 0$ and $H(u) = 0$ otherwise.

Gradients in the input layer

$$\frac{\partial e(\widetilde{\mathbf{x}}, \widetilde{\mathbf{z}})}{\partial \dot{B}_{ij}} = \sum_{\ell=1}^{4} \frac{\partial e^{(\ell)}}{\partial \dot{B}_{ij}} \quad i = 1, \ldots, d, \quad j = 1, \ldots, q. \tag{43}$$

$$\frac{\partial e^{(\ell)}}{\partial \dot{B}_{ij}} = \frac{\partial e^{(\ell)}}{\partial y_j^{(\ell)}} \frac{\partial y_j^{(\ell)}}{\partial \dot{B}_{ij}} + \frac{\partial e^{(\ell)}}{\partial y_j^{(4-\ell+1)}} \frac{\partial y_j^{(4-\ell+1)}}{\partial \dot{B}_{ij}}$$
$$i = 1, \ldots, d, \quad j = 1, \ldots, q, \quad \ell = 1, \ldots, 4. \tag{44}$$

Let

$$\begin{cases} \zeta_j^{(\ell)} & \triangleq & \dfrac{\partial e^{(\ell)}}{\partial y_j^{(\ell)}} \\ \eta_j^{(\ell)} & \triangleq & \dfrac{\partial e^{(\ell)}}{\partial y_j^{(4-\ell+1)}} \end{cases} \quad j = 1, \ldots, q, \quad \ell = 1, \ldots, 4. \tag{45}$$

Using (37) and (45), Eq. (44) can thus be rewritten as:

$$\frac{\partial e^{(\ell)}}{\partial \dot{B}_{ij}} = \zeta_j^{(\ell)} \left(H(\dot{B}_{ij}) x_i^{(\ell)} + [1 - H(\dot{B}_{ij})] x_i^{(4-\ell+1)} \right) + \eta_j^{(\ell)} \left(H(\dot{B}_{ij}) x_i^{(4-\ell+1)} + [1 - H(\dot{B}_{ij})] x_i^{(\ell)} \right). \tag{46}$$

The quantities $\zeta_j^{(\ell)}$ and $\eta_j^{(\ell)}$, computed in the hidden layer, are similar in their definition to the $\delta_k^{(\ell)}$ computed in the output layer. Their computation, which is not as straigthforward, makes use of the chain rule for partial derivatives:

$$\zeta_j^{(\ell)} = \sum_{k=1}^{d} \frac{\partial e^{(\ell)}}{\partial z_k^{(\ell)}} \frac{\partial z_k^{(\ell)}}{\partial y_j^{(\ell)}} = \sum_{k=1}^{d} \delta_k^{(\ell)} \ddot{B}_{kj}^{+} \quad j = 1, \ldots, q, \quad \ell = 1, \ldots, 4. \tag{47}$$

One can see that $\zeta_j^{(\ell)}$ is a weighted sum of the $\delta_k^{(\ell)}$ of the output units. Because the $\delta_k^{(\ell)}$'s must be computed before the $\zeta_j^{(\ell)}$, the process starts from the output layer and works backward to the input layer, hence the name of *backpropagation*. Thanks to this particular ordering of the computations, the complexity of the algorithm is limited to $O(dq)$, which makes the backpropagation very attractive.

Similarly, we can compute:

$$\eta_j^{(\ell)} = \sum_{k=1}^{d} \frac{\partial e^{(\ell)}}{\partial z_k^{(\ell)}} \frac{\partial z_k^{(\ell)}}{\partial y_j^{(4-\ell+1)}} = \sum_{k=1}^{d} \delta_k^{(\ell)} \ddot{B}_{kj}^{-} \quad j = 1, \ldots, q, \quad \ell = 1, \ldots, 4. \tag{48}$$

Finally, coming back to our first notation:

$$\frac{\partial e^{(\ell)}}{\partial B_{ij}} = \frac{\partial e^{(\ell)}}{\partial \dot{B}_{ij}} + \frac{\partial e^{(\ell)}}{\partial \ddot{B}_{ij}} \quad i = 1, \ldots, d, \quad j = 1, \ldots, q, \quad \ell = 1, \ldots, 4. \tag{49}$$

References

[1] T. W. Anderson. *An introduction to multivariate statistical analysis.* Wiley, New-York, 1984.

[2] P. Baldi and K. Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2:53–58, 1989.

[3] P. Baldi and K. Hornik. Learning in linear neural networks: A survey. *IEEE Transactions on Neural Networks*, 6(4):837–858, 1995.

[4] C. Bertoluzza, M. A. Gil, and D. A. Ralescu, editors. *Statistical modeling, analysis and management of fuzzy data.* Physica-Verlag, Heidelberg, 2002.

[5] J. C. Bezdek, J. Keller, R. Krishnapuram, and N. R. Pal. *Fuzzy models and algorithms for pattern recognition and image processing.* Kluwer Academic Publishers, Boston, 1999.

[6] E. Bingham and Heikki Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Proc. 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 2001)*, pages 245–250, F. Provost and R. Srikant (Eds), 2000.

[7] H.-H. Bock and E. Diday, editors. *Analysis of symbolic data. Exploratory methods for extracting statistical information from complex data.* Springer Verlag, Berlin, 2000.

[8] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59:291–294, 1988.

[9] J. J. Buckley and Y. Hayashi. Fuzzy neural networks: A survey. *Fuzzy sets and systems*, 66:1–13, 1994.

[10] P. Cazes, A. Chouakria, E. Diday, and Y. Schektman. Extension de l'analyse en composantes principales à des données de type intervalle. *Revue de Statistique Appliquée*, 14(3):5–24, 1997.

[11] M. Delgado, M. A. Vila, and W. Voxman. On a canonical representation of fuzzy numbers. *Fuzzy sets and systems*, 93:125–135, 1998.

[12] T. Denœux and M. Masson. Multidimensional scaling of interval-valued dissimilarity data. *Pattern Recognition Letters*, 21:83–92, 2000.

[13] P. Diamond and H. Tanaka. Fuzzy regression analysis. In R. Slowinski, editor, *Fuzzy sets in decision analysis, operations research and statistics*, pages 349–387. Kluwer Academic Publishers, Boston, 1998.

[14] D. Dubois, E. Kerre, R. Mesiar, and H. Prade. Fuzzy interval analysis. In D. Dubois and H. Prade, editors, *Fundamentals of Fuzzy sets*, pages 483–581. Kluwer Academic Publishers, Boston, 2000.

[15] D. Dubois, H. T. Nguyen, and H. Prade. Possibility theory, probability and fuzzy sets: Misunderstandings, bridges and gaps. In D. Dubois and H. Prade, editors, *Fundamentals of Fuzzy sets*, pages 343–438. Kluwer Academic Publishers, Boston, 2000.

[16] D. Dubois and H. Prade. *Possibility Theory: An approach to computerized processing of uncertainty.* Plenum Press, New-York, 1988.

[17] K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, Boston, 1990.

[18] J. Gebhardt, M. A. Gil, and R. Kruse. Fuzzy set-theoretic methods in statistics. In R. Slowinski, editor, *Fuzzy sets in decision analysis, operations research and statistics*, pages 311–347. Kluwer Academic Publishers, Boston, 1998.

[19] R. J. Hathaway, J. C. Bezdek, and W. Pedrycz. A parametric model for fusing heterogeneous fuzzy data. *IEEE Transactions on Fuzzy Systems*, 4(3):270–281, 1996.

[20] Y. Hayashi, J.J. Buckley, and E. Czogala. Fuzzy neural network with fuzzy signals and weights. *International J. Intelligent Systems*, 8:527–537, 1993.

[21] S. Heilpern. Representation and application of fuzzy numbers. *Fuzzy sets and systems*, 91:259–268, 1997.

[22] H. Ishibuchi, R. Fujioka, and H. Tanaka. An architecture of neural network for input vectors of fuzzy numbers. In *Proc. FUZZ-IEEE'92*, pages 643–650. IEEE, 1992.

[23] H. Ishibuchi, K. Kwon, and H. Tanaka. A learning algorithm of fuzzy neural networks with triangular fuzzy weights. *Fuzzy sets and systems*, 71:277–293, 1995.

[24] H. Ishibuchi, K. Morioka, and I. B. Turksen. Learning by fuzzified neural networks. *International Journal of Approximate Reasoning*, 13:327–358, 1995.

[25] J. E. Jackson. *A user's guide to principal components.* Wiley, New-York, 1991.

[26] G. J. Klir and M. J. Wierman. *Uncertainty-Based Information. Elements of Generalized Information Theory.* Springer-Verlag, New-York, 1998.

[27] C.N. Lauro and F. Palumbo. Principal component analysis of interval data: a symbolic data analysis approach. *Computational statistics*, 15(1):73–87, 2000.

[28] Z. Li, V. Kecman, and A. Ichikawa. Fuzzified neural network based on fuzzy number operations. *Fuzzy Sets and Systems*, 130:291–304, 2002.

[29] S.T. Liu and C. Kao. Fuzzy measures for correlation coefficient of fuzzy numbers. *Fuzzy Sets and Systems*, 128:267–275, 2002.

[30] M.-H. Masson and T. Denœux. Multidimensional scaling of fuzzy dissimilarity data. *Fuzzy Sets and Systems*, 128(3):339–352, 2002.

[31] W. Pedrycz, J. C. Bezdek, R. J. Hathaway, and G. W. Rogers. Two nonparametric models for fusing heterogeneous fuzzy data. *IEEE Transactions on Fuzzy Systems*, 6(3):411–425, 1998.

[32] O. Rodriguez, E. Diday, and S. Winsberg. Generalization of principal components analysis to histogram data. In *Principles and Practice of knowledge discovery in databases*, Lyon, 2000.

[33] S. Roweis. EM algorithms for PCA and SPCA. In *Neural Information Processing Systems 10*, pages 626–632, 1997.

[34] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the microstructure of Cognition*, volume 1, pages 318–362. MIT Press, Cambridge, 1988.

[35] D. Valentin, H. Abdi, A. J. O'Toole, and G. W. Cottrell. Connectionist models of face processing: A survey. *Pattern Recognition*, 27:1209–1230, 1994.

[36] R. Viertl. *Statistical methods for non-precise data*. CRC Press, New-York, 1996.

[37] L. A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978.

Tables

Table 1: Student dataset.

	M1	M2	P1	P2
Tom	15	fairly good	unknown	[14,16]
David	9	good	fairly good	10
Bob	6	[10,11]	[13,20]	good
Jane	fairly good	very good	19	[10,12]
Joe	very bad	fairly bad	[10,14]	[14]
Jack	1	[4,6]	9	[6,9]

Table 2: Connection weights B_{ij} between the two principal axes and the four original variables, for the Student dataset.

	axis 1	axis 2
M1	0.80	-0.07
M2	0.57	0.17
P1	0.00	0.95
P2	0.08	0.13

Table 3: Reconstruction errors for the artificial data.

Tom	David	Bob	Jane	Joe	Jack
20.97	13.00	10.93	10.66	7.65	14.22

Table 4: Connection weights B_{ij} between the two principal axes and the four original variables, for the augmented Student dataset, with error functions E (columns 1 and 2) and E_a (columns 3 and 4).

	E		E_a	
	axis 1	axis 2	axis 1	axis 2
M1	0.47	0.00	0.83	0.00
M2	0.73	0.06	0.45	0.19
P1	0.00	0.98	0.00	0.90
P2	0.46	-0.08	0.00	0.05

Table 5: Connection weights B_{ij} between the two principal axes and the five original variables, for the sensory evaluation dataset

	axis 1	axis 2
x_1	-0.17	-0.16
x_2	0.92	0.00
x_3	0.32	-0.17
x_4	0.01	0.97
x_5	-0.13	-0.01

Figures

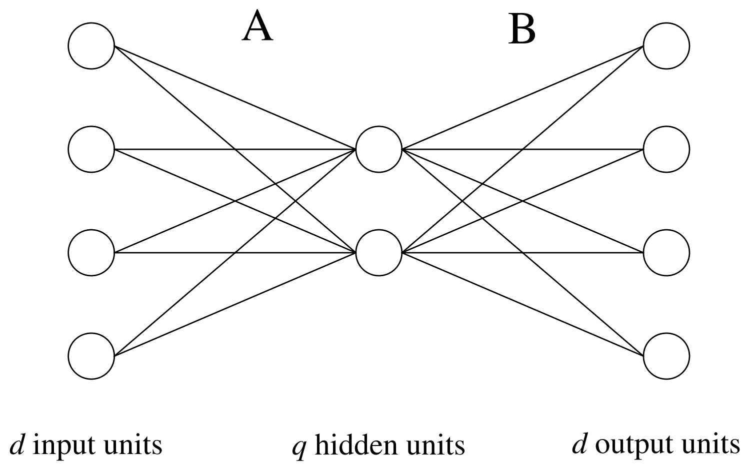

Figure 1: Architecture of the autoassociative network.

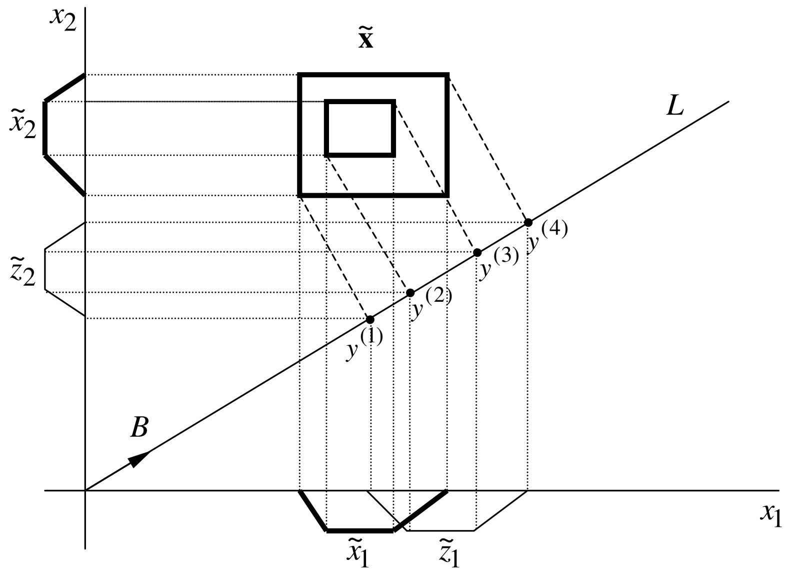

Figure 2: Geometric interpretation of fuzzy PCA in two dimensions ($d = 2$) with one hidden unit ($q = 1$). A vector of fuzzy numbers $\widetilde{\mathbf{x}} = (\widetilde{x}_1, \widetilde{x}_2)'$ is projected onto the line $\mathcal{L}$ directed by the vector B of weights connecting the hidden unit to the two output units, yielding a fuzzy number $\widetilde{y} = (y^{(1)}, y^{(2)}, y^{(3)}, y^{(4)})$ on $\mathcal{L}$. This fuzzy number is then projected back on the two initial axes x_1 and x_2, yielding a reconstructed output vector $\widetilde{\mathbf{z}} = (\widetilde{z}_1, \widetilde{z}_2)'$. The weight vector B is determined to minimize a measure of discrepancy between $\widetilde{\mathbf{x}}$ and $\widetilde{\mathbf{z}}$.

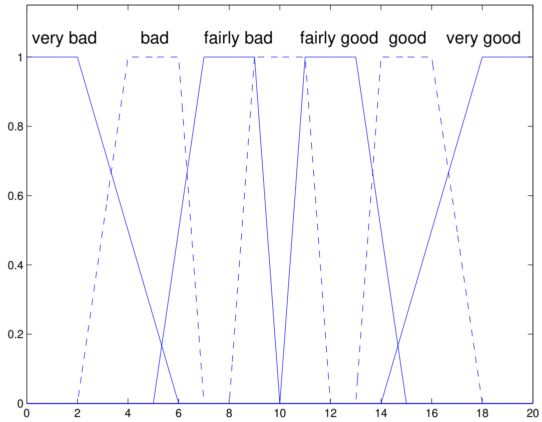

Figure 3: Representation of fuzzy marks.

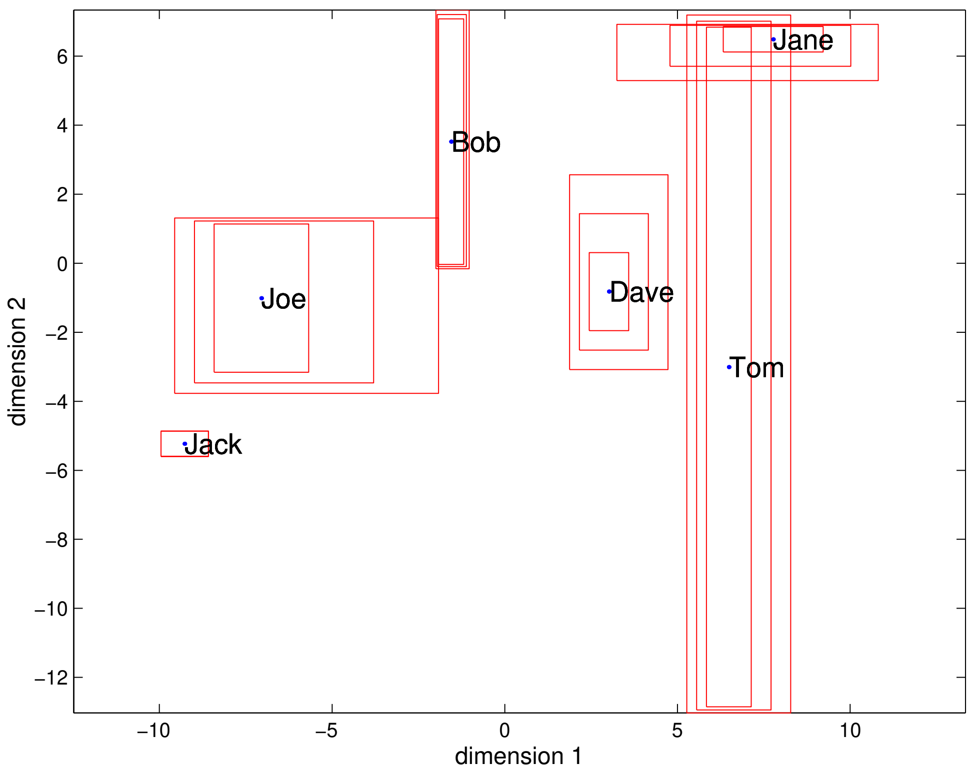

Figure 4: Two-dimensional projection of students dataset using a neural network (supports, cores, and 0.5-cuts).

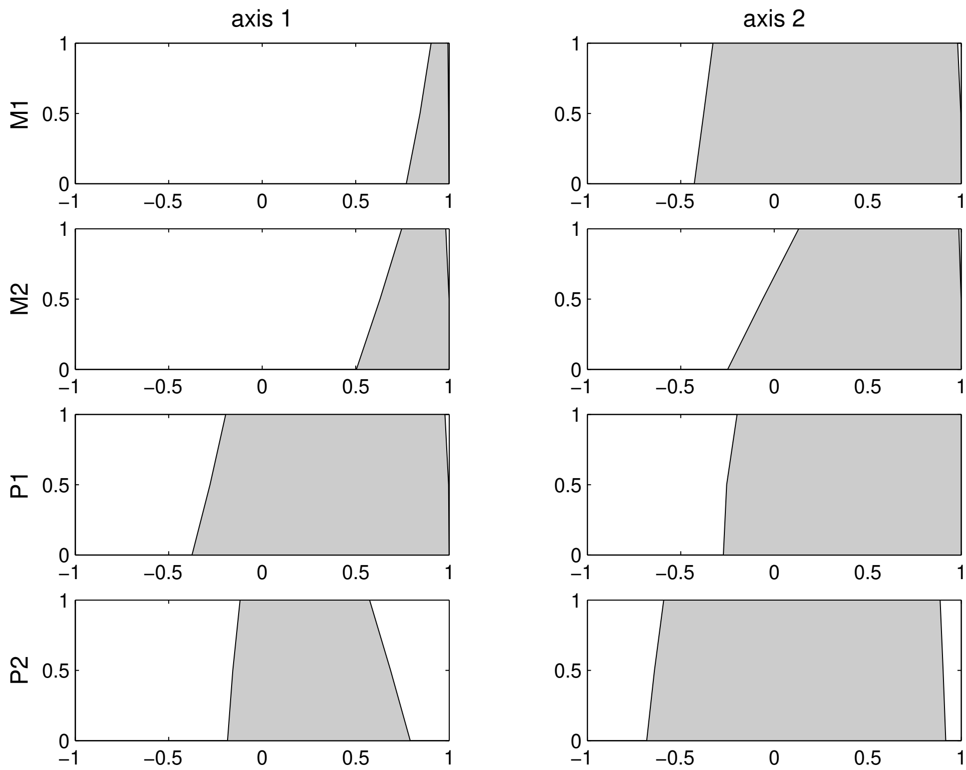

Figure 5: Fuzzy correlation coefficients between the two principal axes and the four original variables for the Students dataset.

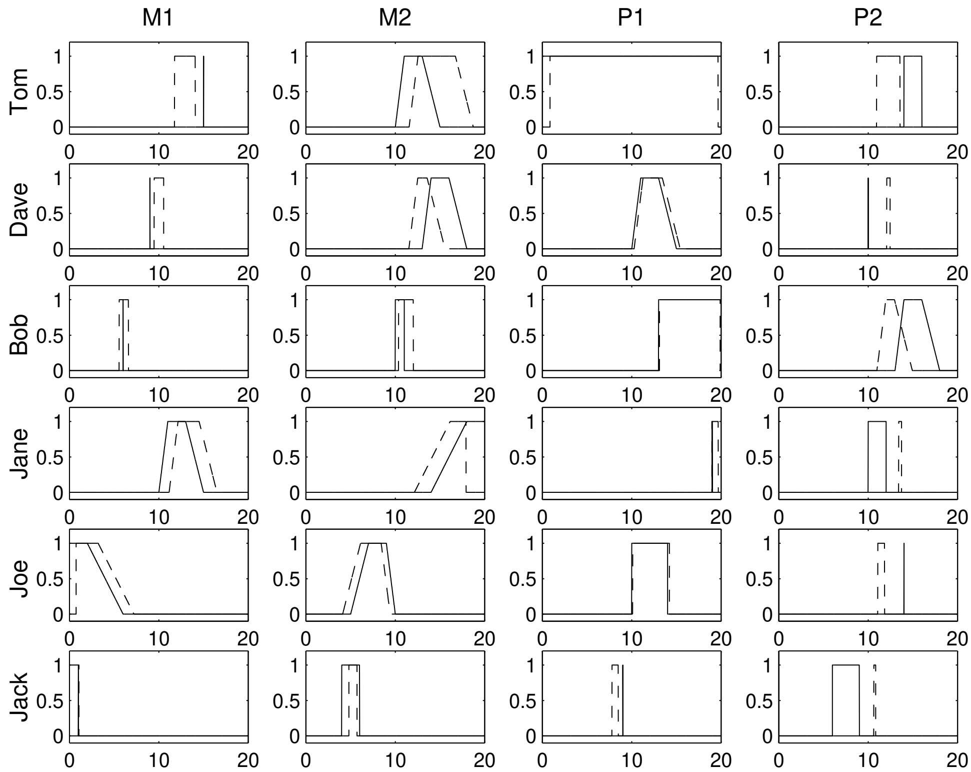

Figure 6: Students dataset: input data (solid lines) and reconstructed data (dashed lines).

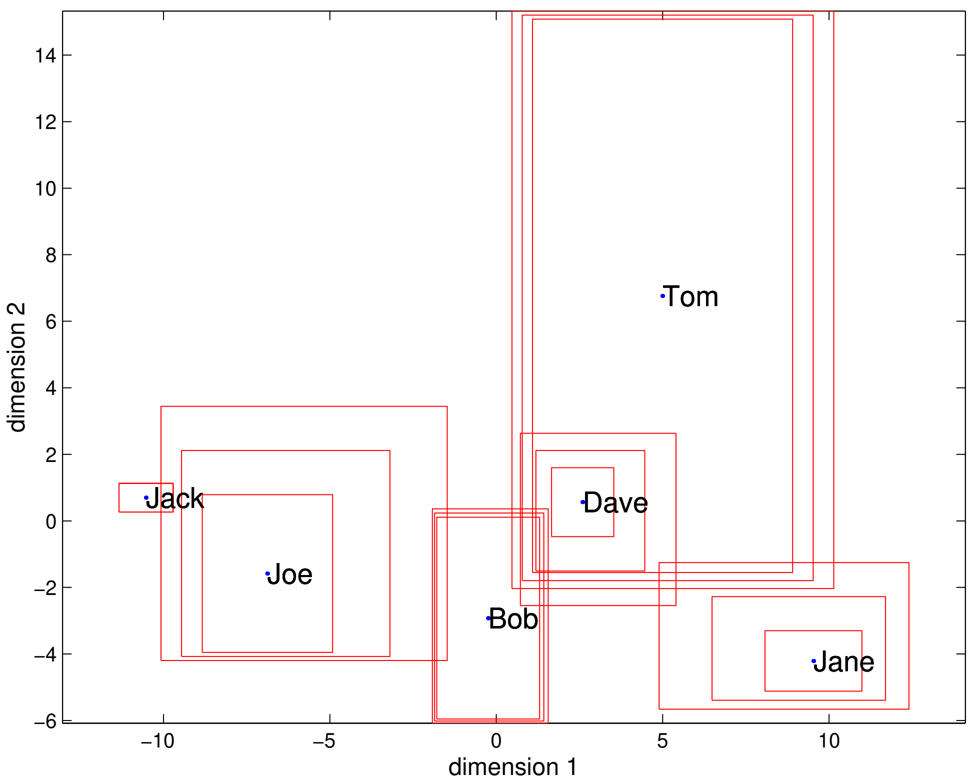

Figure 7: Two-dimensional projection of students dataset using Cazes' centers method [10] (supports, cores, and 0.5-cuts).

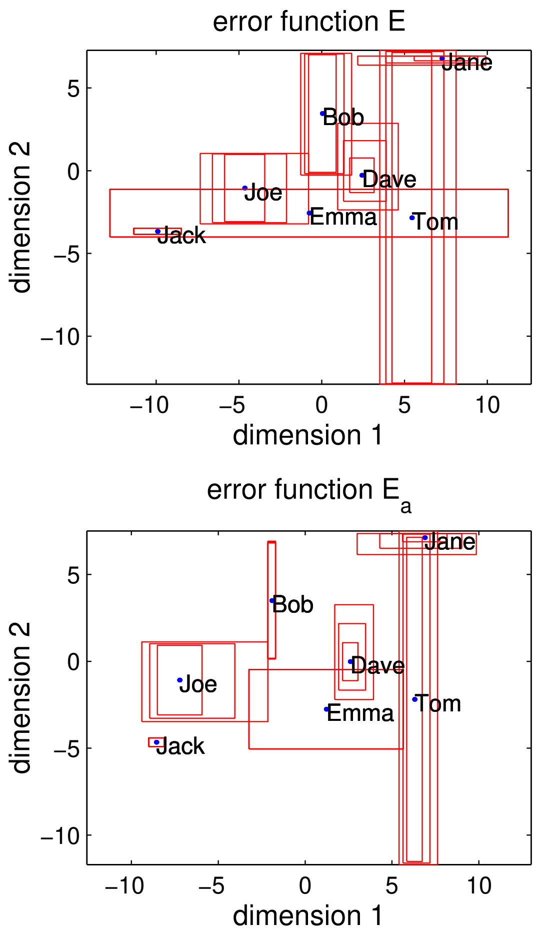

Figure 8: Two-dimensional projection of students dataset (supports, cores, and 0.5-cuts), using error functions E (top) and E_a (bottom).

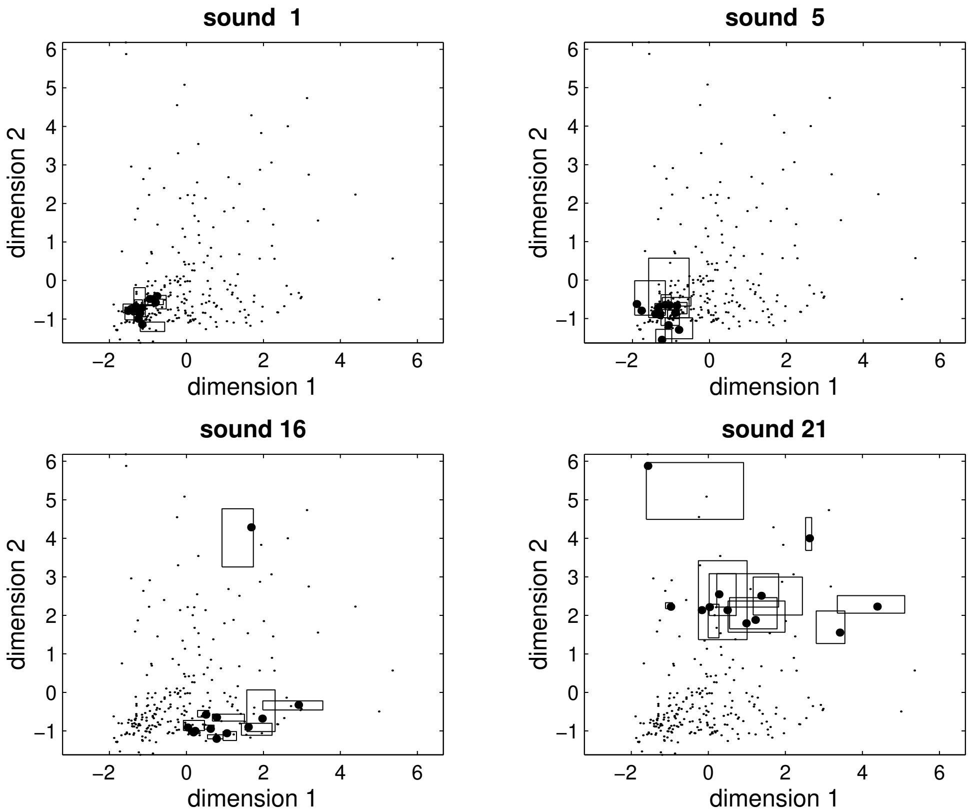

Figure 9: Examples of two-dimensional projections of sounds. Each picture shows how a given sound is perceived by the twelve assessors. Solid lines and plain circles represent, respectively, the 0.5-cuts and cores of the projections. Thin points represent the centroid projections for the other pairs (sound,assessor).

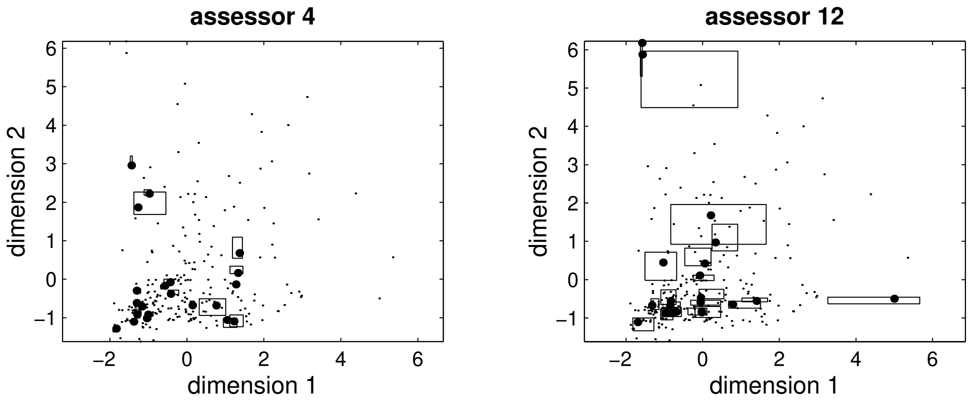

Figure 10: Examples of two-dimensional projections of sounds. Each picture shows how a given assessor perceives the twenty one sounds. Solid lines and plain circles represent, respectively, the 0.5-cuts and cores of the projections. Thin points represent the centroid projections for the other pairs (sound,assessor).

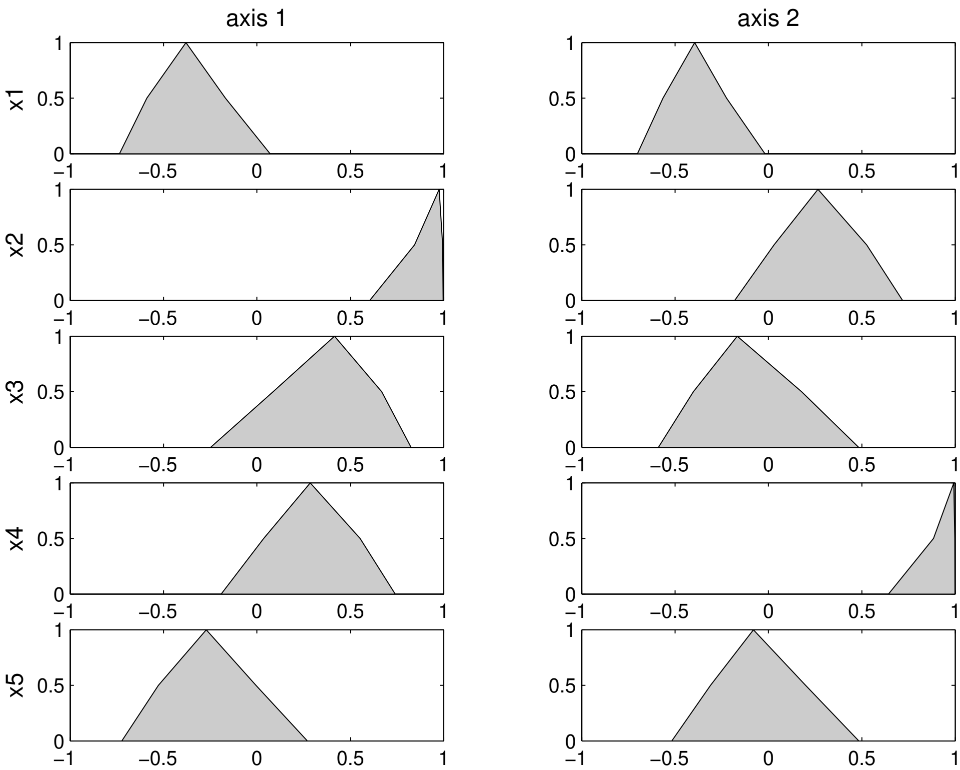

Figure 11: Fuzzy correlation coefficients between the two principal axes and the five original variables (sensory evaluation data).

Vitae

T. Denœux graduated in 1985 as an engineer from the Ecole Nationale des Ponts et Chaussées in Paris, and received a doctorate from the same institution in 1989. He is currently a Full Professor with the Department of Information Processing Engineering at the Université de Technologie de Compiègne, France. His current research interests concern fuzzy data analysis, belief functions theory and, more generally, the management of imprecision and uncertainty in data analysis, pattern recognition and information fusion.

Marie-Hélène Masson received the Engineer degree in Computer Science and a PhD from the Université de Technologie de Compiègne. She has been an assistant professor at the Université de Picardie Jules Verne (IUT de l'Oise) and a member of the Heudiasyc laboratory (UMR CNRS 6599) since 1993. Her research interests include statistical pattern recognition and data analysis.