Granular self-organizing map (grSOM) for structure identification

Vassilis G. Kaburlasos and S. E. Papadakis

Department of Industrial Informatics
Division of Computing Systems
Technological Educational Institution of Kavala
GR- 65404 Kavala, Greece

Contact Author

Name: **Vassilis G. Kaburlasos**

Postal Address: **Department of Industrial Informatics**
Technological Educational Institution of Kavala
GR 654 04 Kavala
Greece

Phone Number: **+30 (2510) 462-320**

Fax Number: **+30 (2510) 462-348**

E-mail Address: **vgkabs@teikav.edu.gr**

Granular self-organizing map (grSOM) for structure identification

Abstract

This work presents a useful extension of Kohonen's Self-Organizing Map (KSOM) for structure identification in linguistic (fuzzy) system modeling applications. More specifically the *granular SOM* neural model is presented for inducing a distribution of nonparametric fuzzy interval numbers (*FIN*s) from the data. A *FIN* can represent a local probability distribution function and/or a conventional fuzzy set; moreover a *FIN* is interpreted as an *information granule*. Learning is based on a novel metric distance $d_K(.,.)$ between *FIN*s. The metric $d_K(.,.)$ can be tuned nonlinearly by a mass function $m(x)$, the latter attaches a weight of significance to a real number 'x' in a data dimension. Rigorous analysis is based on mathematical lattice theory. A *grSOM* can cope with ambiguity by processing linguistic (fuzzy) input data and/or intervals. This work presents a simple *grSOM* variant, namely *greedy grSOM*, for classification. A genetic algorithm (GA) introduces tunable nonlinearities during training. Extensive comparisons are shown with related work from the literature. The practical effectiveness of the *greedy grSOM* is demonstrated comparatively in three benchmark classification problems. Statistical evidence strongly suggests that the proposed techniques improve classification performance. In addition, the *greedy grSOM* induces descriptive decision-making knowledge (fuzzy rules) from the training data.

Keywords: Self organizing map (SOM); linguistic granule; classification; fuzzy interval number (FIN); fuzzy inference system (FIS); lattice theory; statistics; genetic algorithm (GA) optimization

List of symbols

$[a,b]^h$	a generalized interval of height h
CALFIN	an algorithm for constructing a *FIN*
CDF	Cumulative Distribution Function
cone(X)	the cone generated in a linear space by the set X
conv(X)	the convex hull of the set X
$d(x,y)$	a metric distance between x and y

$d_h([a,b]^h,[c,d]^h)$	a metric distance between generalized intervals $[a,b]^h$ and $[c,d]^h$
$d_K(F_1,F_2)$	a metric distance between *FIN*s F_1 and F_2
$d_p(A,B)$	Minkowski distance between N-tuples of *FIN*s A and B for p=1,2,…
e_1^h, e_2^h	basis vectors in M^h
f_h	a strictly increasing function
$F(h)$	a generalized interval of height h
F	metric lattice of *FIN*s
F_+	the set of positive *FIN*s
F_0	the set of trivial (positive) *FIN*s
F_-	the set of negative *FIN*s
FIN	Fuzzy Interval Number
F_a	a-cut of a fuzzy set F
KSOM	Kohonen's Self-Organizing Map
L	a general mathematical lattice
M	the set of all generalized intervals of height $h\in(0,1]$
M^h	the metric lattice of generalized intervals of height h
M_+^h	the set of positive generalized intervals of height h
M_0^h	the set of trivial (positive) generalized intervals of height h
M_-^h	the set of negative generalized intervals of height h
$m(x)$	a mass function
$\mu_{a,b}^h(x)$	membership function of generalized interval $[a,b]^h$
R	the totally-ordered lattice of real numbers
R^+	the set of positive real numbers
R_0^+	the set of positive real numbers including zero (0)
SOM	Self-Organizing Map
$v(x)$	a positive valuation function, where x is a lattice element
$v_h(.)$	a positive valuation function in lattice M^h
$\leq$	lattice ordering relation
$\|\|$	lattice incomparability operator
$\vee$	lattice join operator
$\wedge$	lattice meet operator
$\bigcup$	set union operator
$\subseteq$	set inclusion operator

1. Introduction

A popular neural network for clustering is Kohonen's self-organizing map (KSOM) devised mainly for visualization of nonlinear relations of multi-dimensional data (Kohonen, 1995). The KSOM is known for its effectiveness in real world applications (Haritopoulos, Yin, & Allinson, 2002; Kohonen *et al*, 2000; Papadimitriou *et al*, 2001; Principe, Wang, & Motter, 1998). Variants of KSOM have been reported for learning nonvectorial data (Cottrell, Ibbou, & Letrémy, 2004; Kohonen, 1996; Kohonen & Somervuo, 1998; Seo & Obermayer, 2004; Somervuo, 2004). This work investigates the potential of KSOM towards structure identification in linguistic (fuzzy) system modeling applications. More specifically, the emphasis of this work is on inducing descriptive decision-making knowledge (fuzzy rules) from the training data for classification. Our contribution is summarized as follows.

Based on sound mathematics this work proposes an extension of KSOM, namely *granular SOM* or *grSOM* for short, for inducing a distribution of *Fuzzy Interval Numbers* (*FIN*s) from data samples. A *FIN* can represent, nonparametrically, a local probability distribution and/or a fuzzy set. Learning is driven by a tunable metric distance $d_K(.,.)$ between *FIN*s. An integrable *mass* function $m(x)$ can nonlinearly tune metric $d_K(.,.)$ by attaching a weight of significance to a real number 'x' in a data dimension. A genetic algorithm (GA) computes optimally mass function $m(x)$. The objective of a *grSOM* model in the context of this work is linguistic (fuzzy) rule induction towards classification rather than data visualization.

We use the name *grSOM* to denote a family of KSOM-based architectures, which process *FIN*s. A simple *grSOM* variant, namely *greedy grSOM*, is presented as a fuzzy neural network for classification. More specifically, the *greedy grSOM* induces nonparametric *FIN*s from the training data such that a *FIN* specifies a fuzzy data cluster. The induced *FIN*s are interpreted as antecedents (IF parts) of linguistic (fuzzy) rules; the corresponding rule consequents (THEN parts) are category labels.

This paper builds on a body of work reported lately (Kaburlasos, 2002, 2004; Kaburlasos & Papadakis, 2004; Kaburlasos & Petridis, 2003; Petridis & Kaburlasos, 2003). Substantial novelties are introduced here as explained below. Rigorous mathematical analysis is based on lattice theory. More specifically, lattices of *generalized intervals* are used for introducing a tunable metric distance $d_K(.,.)$ between *FIN*s. Sound evidence, including both theoretical and experimental evidence, is presented regarding advantages of the *greedy grSOM* in linguistic (fuzzy) classification applications.

The layout of this paper is as follows. Section 2 outlines the problem of structure identification in linguistic (fuzzy) system modeling applications. Section 3 summarizes the mathematical background. Section 4 considers an algorithm for constructing a *FIN*, moreover interpretations for a *FIN* are presented. Section 5 details the *greedy granular Self-Organizing*

Map (*greedy grSOM*) algorithm. Section 6 shows comparisons with related work from the literature; the novelties of this work are also summarized. Experimental results are presented in section 7 including a discussion. Section 8 concludes by summarizing the contribution as well as potential future extensions of this work.

2. The problem of structure identification

This section outlines the problem of structure identification in linguistic (fuzzy) system modeling. A fuzzy inference system (FIS) includes a knowledge-base of fuzzy rules. For instance, Fig.1 shows a typical "Mamdani type" FIS, involving triangular fuzzy membership functions, where the antecedent (IF part) of a rule is the conjunction of N fuzzy statements and the consequent (THEN part) of a rule is a single fuzzy statement (Mamdani & Assilian, 1975). More specifically, Fig.1 displays L fuzzy rules R1,…,RL. An input vector $x=(x_1,\ldots,x_N)\in\mathsf{R}^N$ activates in parallel all L rules by a *fuzzification* procedure. The fuzzy rule consequents are combined and, finally, a single number is produced by a *defuzzification* procedure. A Mamdani type FIS could include other fuzzy membership function shapes than triangular ones. A different type FIS is obtained using an algebraic expression $y=f(x_1,\ldots,x_N)$, e.g. a linear expression $y=a_1x_1+\ldots+a_Nx_N$, as a consequent to a rule; hence a "Tagaki-Sugeno-Kang (TSK) type" FIS results (Tagaki & Sugeno, 1985).

It turns out that a FIS is a mechanism for implementing a function f: $\mathsf{R}^N\rightarrow\mathsf{K}$, where the range K may be either discrete or continuous; in particular the Mamdani type FIS in Fig.1 implements a function f: $\mathsf{R}^N\rightarrow\mathsf{R}$. Another popular function, which a FIS can implement is a classifier (Ishibuchi, Nakashima, & Murata, 1999; Mitra & Hayashi, 2000; Mitra & Pal, 1994), where the range K of function f: $\mathsf{R}^N\rightarrow\mathsf{K}$ is a discrete set of class labels.

A FIS is called *linguistic* system model because a linguistic label such as "high", "low", "average", etc. is attached to a fuzzy set. *Structure identification* concerns the important problem of placing optimally, in a sense, fuzzy sets on the real line for all rules involved in a FIS (Sugeno & Kang, 1988). Both the location and the membership function shapes of corresponding fuzzy sets are induced from the data/measurements; note that usually simple parametric fuzzy membership functions are used, for example triangular, trapezoidal, and Gaussian membership functions.

A solution to the structure identification problem is typically pursued using various clustering and/or supervised learning algorithms including (fuzzy) neural networks (Jang & Sun, 1995; Kecman, 2001; Leng, McGinnity, & Prasad, 2005; Mitra & Hayashi, 2000; Papadakis & Theocharis, 2002; Pomares *et al*, 2002; Setnes, 2000; Tagaki & Sugeno, 1985). In particular, Lin & Lin (1997) employ the *fuzzy ART* neural network for computing hyperboxes (clusters)

in R^N. This work proposes an extension of Kohonen's self-organizing map (KSOM) neural network for structure identification with emphasis on pattern classification problems.

3. Mathematical background

Parts of the material in this section have been presented previously in various contexts (Kaburlasos, 2002, 2004; Kaburlasos & Papadakis, 2004; Kaburlasos & Petridis, 2003; Petridis & Kaburlasos, 2003). This section presents concisely a unified enhancement including, in addition, new material as pointed out explicitly on occasion. The mathematics presented in this section will be necessary for introducing novel techniques below.

3.1 Metric lattices M^h of generalized intervals

A *generalized interval* (positive, or negative) *of height h* is defined in the following.

Definition 1

A *positive generalized interval of height h* is a mapping $\mu^h_{x_1,x_2}(x): \mathsf{R}\rightarrow\{0,h\}$ (where $x_1\leq x_2$ and $h\in(0,1]$) given by $\mu^h_{x_1,x_2}(x)=\begin{cases}h, & x_1\leq x\leq x_2\\ 0, & otherwise\end{cases}$;

a *negative generalized interval of height h* is a mapping $\mu^h_{x_1,x_2}(x): \mathsf{R}\rightarrow\{0,-h\}$ (where $x_1>x_2$ and $h\in(0,1]$) given by $\mu^h_{x_1,x_2}(x)=\begin{cases}-h, & x_1\geq x\geq x_2\\ 0, & otherwise\end{cases}$.

■

Definition 1 above is a simplified *functional* equivalent of a previous definition (Kaburlasos, 2004). A generalized interval is a simple box function, either positive or negative. No specific interpretation is attached to generalized interval. Generalized intervals will be useful below for introducing a metric distance between *fuzzy numbers*, where a fuzzy number is defined as an interval-supported convex fuzzy set of height one. A generalized interval will be denoted more compactly as $[x_1,x_2]^h$, where $x_1\leq x_2$ for positive- and $x_1>x_2$ for negative- generalized intervals. A positive generalized interval $[x_1,x_1]^h$ will be called, in particular, *trivial (positive) generalized interval*. Previous work has considered, as well, trivial negative generalized intervals (Kaburlasos, 2004). In the interest of simplicity, and without loss of generality, this work considers only positive trivial generalized intervals.

The set of positive (negative) generalized intervals of height h will be denoted by M^h_+ (M^h_-); in particular, the set of trivial positive generalized intervals will be denoted by M^h_0, where $\mathsf{M}^h_0\subset\mathsf{M}^h_+$. The set of generalized intervals of height h will be denoted by M^h, i.e. $\mathsf{M}^h=$

$\mathsf{M}_-^h \cup \mathsf{M}_+^h$. The set-union $\bigcup_{h\in(0,1]} \mathsf{M}^h$ is the set M of *generalized intervals*, symbolically $\mathsf{M}=\bigcup_{h\in(0,1]} \mathsf{M}^h$. Our interest here is in generalized intervals $[x_1,x_2]^h$ with $h\in(0,1]$ because the latter may emerge from the *a*-cuts of conventional fuzzy numbers. More specifically, to an *a*-cut $F_a = \{x: x\geq a\}$ of a fuzzy number F there corresponds a positive generalized interval with support F_a and height '*a*'. An *ordering* relation can be defined on $\mathsf{M}^h \times \mathsf{M}^h$ as follows.

(R1) If $[a,b]^h, [c,d]^h \in \mathsf{M}_+^h$ then: $[a,b]^h \leq [c,d]^h \Leftrightarrow [a,b] \subseteq [c,d]$,

(R2) If $[a,b]^h, [c,d]^h \in \mathsf{M}_-^h$ then: $[a,b]^h \leq [c,d]^h \Leftrightarrow [d,c] \subseteq [b,a]$, and

(R3) If $[a,b]^h \in \mathsf{M}_-^h, [c,d]^h \in \mathsf{M}_+^h$ then: $[a,b]^h \leq [c,d]^h \Leftrightarrow [b,a] \cap [c,d] \neq \varnothing$.

The partial ordering relation above has been introduced elsewhere (Kaburlasos, 2002, 2004), nevertheless a simplified notation is used here. Rule (R1) indicates that a positive generalized interval $[a,b]^h$ is smaller than another one $[c,d]^h$ if and only if the interval support of $[a,b]^h$ is included in the interval support of $[c,d]^h$. Rule (R2) indicates, dually, the converse for negative generalized intervals. Finally, rule (R3) indicates that a negative generalized interval $[a,b]^h$ is smaller than a positive one $[c,d]^h$ if and only if the corresponding interval supports overlap. In all other cases generalized intervals $[a,b]^h$ and $[c,d]^h$ are *incomparable*, symbolically $[a,b]^h \| [c,d]^h$.

The aforementioned ordering relation is a *partial ordering* relation[1]; moreover, the partially ordered set M^h of generalized intervals is a *mathematical lattice*[2] (Kaburlasos, 2002, 2004; Petridis & Kaburlasos, 2003). Note that a lattice ordering is a partial ordering but not vice versa. Useful functions can be defined in a general lattice L as shown next.

Definition 2

A *valuation* in a lattice L is a real function v: $\mathsf{L}\rightarrow\mathsf{R}$ which satisfies $v(x)+v(y)=v(x\vee y)+v(x\wedge y)$, $x,y\in\mathsf{L}$. A valuation is called *positive* if and only if $x<y$ in L implies $v(x)<v(y)$ for $x,y\in\mathsf{L}$.

■

A positive valuation $v(.)$ in a lattice L implies a *metric distance*[3] function d: $\mathsf{L}\times\mathsf{L}\rightarrow\mathsf{R}_0^+$ given by $d(x,y)=v(x\vee y)-v(x\wedge y)$ for $x,y\in\mathsf{L}$ (Birkhoff, 1967). A positive valuation function v_h: $\mathsf{M}^h\rightarrow\mathsf{R}$ can be defined in lattice M^h as follows.

---

[1] A *partial ordering* relation, symbolically $\leq$, in a set $\mathcal{S}$ has to be: (PO1) *reflexive* ($x\leq x$), (PO2) *antisymmetric* ($x\leq y$ and $y\leq x$ imply $x=y$), and (PO3) *transitive* ($x\leq y$ and $y\leq z$ imply $x\leq z$), where $x,y,z\in\mathcal{S}$.

[2] A *mathematical lattice* is a partially ordered set L any two of whose elements have a greatest lower bound (g.l.b.) or "meet" denoted by $x\wedge y$, and a least upper bound (l.u.b.) or "join" denoted by $x\vee y$.

[3] A *metric distance* in a set $\mathcal{S}$ is a real function d: $\mathcal{S}\times\mathcal{S}\rightarrow\mathsf{R}$ which satisfies: (MD0) $d(x,y)\geq 0$, $x,y\in\mathcal{S}$, (MD1) $d(x,y)=0 \Leftrightarrow x=y$, $x\in\mathcal{S}$, (MD2) $d(x,y) = d(y,x)$, $x,y\in\mathcal{S}$, - *Symmetry*, and (MD3) $d(x,y) \leq d(x,z) + d(z,y)$, $x,y,z\in\mathcal{S}$ - *Triangle Inequality*.

Proposition 3

Let f_h: $\mathsf{R}\rightarrow\mathsf{R}$ be a strictly increasing function. Then the function v_h: $\mathsf{M}^h\rightarrow\mathsf{R}$ given by

$$v_h([a,b]^h)= f_h(b) - f_h(a)$$

is a *positive valuation* function in lattice M^h. It follows that function d_h: $\mathsf{M}^h\times\mathsf{M}^h\rightarrow\mathsf{R}_0^+$ given by

$$d_h([a,b]^h,[c,d]^h) = [f_h(a\vee c) - f_h(a\wedge c)] + [f_h(b\vee d) - f_h(b\wedge d)]$$

is a metric distance in lattice M^h.

■

The proof of proposition 3 will be shown elsewhere for lack of space.

A strictly increasing function f_h: $\mathsf{R}\rightarrow\mathsf{R}$ can be constructed from an integrable *mass function* m_h: $\mathsf{R}\rightarrow\mathsf{R}_0^+$ as follows: $f_h(x)= \int_0^x m_h(t)\mathrm{d}t$. Note that the latter integral is positive (negative) for $x>0$ ($x<0$). One may regard a mass function $m_h(x)$ as an instrument for attaching "a weight of significance" to a real number x. Various mass functions can be employed; for instance, using mass function $m_h(x)=h$ it follows metric distance $d_h([a,b]^h,[c,d]^h)= h(|a-c|+|b-d|)$ in M^h. The following example demonstrates the calculation of $d_h(.,.)$.

Example 4

Consider the positive generalized intervals $[-1,0]^1$ and $[3,4]^1$. Using mass function $m(x)=1$ it follows $d_1([-1,0]^1,[3,4]^1)= v_1([-1,0]^1\vee[3,4]^1)-v_1([-1,0]^1\wedge[3,4]^1)= v_1([-1,4]^1)-v_1([3,0]^1)= 5+3 = 8$. Fig.2(a) and Fig.2(b) show two different symmetric mass functions $m_1(x)= 3x^2$ and $m_2(x)= \frac{2e^{-x}}{(1+e^{-x})^2}$, respectively. The corresponding strictly increasing functions $f_1(x)$ and $f_2(x)$ are shown, respectively, in Fig.2(c) and Fig.2(d). More specifically, on the one hand, a steeply increasing function f_1 is shown in Fig.2(c), that is cubic function $v_1(x)=x^3$; the computation of metric $d_1(.,.)$ using $m=m_1$ results in $d_1([-1,0]^1,[3,4]^1)= f_1([-1,0]^1\vee[3,4]^1)-f_1([-1,0]^1\wedge[3,4]^1)= 65+27 = 92$. On the other hand, a slowly increasing (saturated) function f_2 is shown in Fig.2(d), that is the logistic function $v_2(x)=\frac{2}{1+e^{-x}}-1$; the computation of metric distance $d_1(.,.)$ using $m=m_2$ results in $d_1([-1,0]^1,[3,4]^1) = f_2([-1,0]^1\vee[3,4]^1)-f_2([-1,0]^1\wedge[3,4]^1) \approx 1.4262 + 09052 \approx 2.3314$.

■

The example above was meant to demonstrate that a different *mass function* could change substantially the distance between two intervals.

3.2 Vector lattices and convexity in $\mathbb{M}^h$

This section shows a useful algebraic structure in metric lattice $\mathbb{M}^h$. Any treatment of convexity is a novelty introduced in this paper. Note that convex computations are essential for self-organizing map (SOM) algorithms.

An element $[a,b]^h$ of $\mathbb{M}^h$, $h\in(0,1]$ has been represented by a pair of real numbers in the Cartesian product space $\mathbb{R}\times\mathbb{R}$. Since the space $\mathbb{R}^2$ is a real linear space, it follows that the space $\mathbb{M}^h$ is, likewise, a *real linear space*. More specifically,

- *addition* in $\mathbb{M}^h$ is defined as $[a,b]^h+[c,d]^h=[a+c,b+d]^h$.
- *multiplication* (by a real number k) in $\mathbb{M}^h$ is defined as $k[a,b]^h=[ka,kb]^h$.

A generalized interval in $\mathbb{M}^h$ is called *vector* of the linear space $\mathbb{M}^h$. We remark that the term "vector" is used here in the standard linear algebra sense (Itô, 1987).

A lattice-ordered linear space, such as space $\mathbb{M}^h$ in this work, is called *vector lattice* or *Riesz space* (Itô, 1987). We remark that a theory of vector lattices has been introduced in Riesz (1928) and further developed by several authors (Vulikh, 1967).

The dimension of linear space $\mathbb{M}^h$, apparently, equals two. A convenient *basis* can be selected in $\mathbb{M}^h$ as follows: $[a,b]^h= [a,a+(b-a)]^h = [a,a]^h + [0,b-a]^h = a[1,1]^h + (b-a)[0,1]^h = ae_1^h + be_2^h$; therefore basis $(e_1^h, e_2^h)=([1,1]^h, [0,1]^h)$ has been selected in $\mathbb{M}^h$, where $a,b\in\mathbb{R}$.

A subset C of a linear space is called *cone* if for all $x\in C$ and $\lambda>0$, we have $\lambda x\in C$ (Bertsekas, Nedic, & Ozdaglar, 2003). Two interesting cones emerge in the linear space $\mathbb{M}^h$ including, first, the cone $\mathbb{M}_+^h$ of positive generalized intervals and, second, the cone $\mathbb{M}_-^h$ of negative generalized intervals. More specifically, using the aforementioned basis $(e_1^h, e_2^h)=([1,1]^h, [0,1]^h)$ it can be easily shown that 1) if $x\in\mathbb{M}_+^h$ and $\lambda>0$ then $\lambda x\in\mathbb{M}_+^h$, and 2) if $x\in\mathbb{M}_-^h$ and $\lambda>0$ then $\lambda x\in\mathbb{M}_-^h$. Powerful theorems for convex analysis and optimization in $\mathbb{R}^n$ (for n=2) become available in $\mathbb{M}^h$ including the following one (Bertsekas, Nedic, & Ozdaglar, 2003).

Theorem 5 (**Caratheodory's Theorem**)

Let X be a nonempty subset of $\mathbb{R}^n$.

(a) Every x in cone(X) can be represented as a positive combination of vectors $x_1,\ldots,x_m$ from X that are linearly independent.

(b) Every x in conv(X) can be represented as a convex combination of vectors $x_1,\ldots,x_m$ from X such that $x_2-x_1,\ldots,x_m-x_1$ are linearly independent.

■

We remark in the above theorem that: 'cone(X)' is the *cone generated by X*, i.e. the set of nonnegative combinations of elements of X; 'conv(X)' is the *convex hull* of the set X, i.e. the intersection of all convex sets containing X; a 'positive combination of vectors $x_1,\ldots,x_m$' is a

linear combination $\sum_{i=1}^{m} a_i x_i$, where $a_i \in \mathsf{R}$ are all positive; a 'convex combination of vectors $x_1,\ldots,x_m$' is a positive combination $\sum_{i=1}^{m} a_i x_i$ such that $\sum_{i=1}^{m} a_i = 1$.

3.3 The metric space F of FINs

Consider the following definition of a *Fuzzy Interval Number* (*FIN*).

Definition 6

A *Fuzzy Interval Number* (*FIN*) is a function F: $(0,1] \rightarrow \mathsf{M}$ such that (1) $F(h) \in \mathsf{M}^h$, (2) either $F(h) \in \mathsf{M}_+^h$ (*positive FIN*) or $F(h) \in \mathsf{M}_-^h$ (*negative FIN*) for all $h \in (0,1]$, and (3) $h_1 \leq h_2 \Rightarrow \{x: F(h_1) \neq 0\} \supseteq \{x: F(h_2) \neq 0\}$, where $0 \leq h_1 \leq h_2 \leq 1$.

■

We remark that condition (3) in the above definition indicates that the interval support of generalized interval $F(h_1)$ is greater-than or equal-to the support of generalized interval $F(h_2)$ for $h_1 \leq h_2$. Definition 6 is an improved simplification of a corresponding definition in Kaburlasos (2004) where the requirement for 'continuity' in Kaburlasos (2004) is dropped here without loss of generality. The set of *FIN*s will be denoted by F. More specifically the set of *positive* (*negative*) *FIN*s will be denoted by F_+ (F_-); in particular, the set of trivial positive *FIN*s will be denoted by F_0, where $\mathsf{F}_0 \subset \mathsf{F}_+$. Fig.3 shows examples of two negative *FIN*s (i.e. *FIN*s F_m, F_n), one trivial *FIN* (i.e. *FIN* F_t), and two positive *FIN*s (i.e. *FIN*s F_q, F_p). We point out that a *FIN* is an abstract mathematical notion. Various interpretations can be proposed for a *FIN*. For instance, on the one hand, a positive *FIN* may be interpreted as a conventional fuzzy number; moreover a statistical interpretation is presented here below for a positive *FIN*. On the other hand, a negative *FIN* may be interpreted as an *intuitionistic (fuzzy) set* in the sense of Atanasoff (Atanasoff, 1999); more specifically the membership function of a negative *FIN* may denote the degree of certainty that a real number does *not* belong to a fuzzy set. The following proposition from Kaburlasos (2004) introduces a metric distance in the set F of Fuzzy Interval Numbers (*FIN*s).

Proposition 7

Let F_1 and F_2 be *FIN*s in F. A *metric distance* function d_K: $\mathsf{F} \times \mathsf{F} \rightarrow \mathsf{R}_0^+$ is given by $d_K(F_1,F_2) = c\int_0^1 d_h(F_1(h),F_2(h))dh$, where c is a positive normalizing coefficient, and $d_h(F_1(h),F_2(h))$ is a metric distance between generalized intervals $F_1(h)$ and $F_2(h)$.

■

We point out that the metric distance $d_K(F_1,F_2)$ between *FIN*s F_1 and F_2 is based on the metric distance $d_h(F_1(h),F_2(h))$ between generalized intervals $F_1(h)$ and $F_2(h)$, where the definition of

the latter is based on both positive and negative generalized intervals. Hence, negative generalized intervals are instrumental for defining $d_K(.,.)$. The computation of the metric distance $d_K(.,.)$ is demonstrated in the following.

Two pairs of *FIN*s (E1,F) and (E2,F) are shown in Fig.4(a) and Fig.4(b), respectively. Note that both *FIN*s E1 and E2 assume their maximum value of 1 at the same point x=5; moreover the *center of gravity* for both *FIN*s E1 and E2 is at x=5. Therefore if a *FIN* is to be represented by a single number, e.g. either the number where a *FIN* attains its maximum value or the corresponding *FIN* center of gravity, then the distance between *FIN*s E1 and F equals the distance between *FIN*s E2 and F. Nevertheless, a more sensible result is obtained using distance $d_K(.,.)$ as illustrated next.

Fig.4(c) plots the metric distances $d_K(E1(h),F(h))$ and $d_K(E2(h),F(h))$, respectively, in solid line and dotted line as functions of h in (0,1]. It might be interesting to point out that both curves in Fig.4(c) have a "break point" at h=0.8, as expected from the break points in the membership functions of both *FIN*s E1 and E2 at h=0.8. The curves in Fig.4(c) show that $d_K(E1(h),F(h)) < d_K(E2(h),F(h))$ for values of h smaller than $h\approx 0.45$, whereas $d_K(E1(h),F(h)) > d_K(E2(h),F(h))$ for larger values of h. The latter inequalities are expected from Fig.4(a) and Fig.4(b) because the "lower" part of *FIN* E2 is further from *FIN* F than the lower part of *FIN* E1, and vice versa for the upper parts. By computing the areas underneath the corresponding curves in Fig.4(c) it turns out $d_K(E1,F)\approx 6.18$ moreover $d_K(E2,F)\approx 5.81$.

The previous results can be extended in the Cartesian product $\mathsf{F}^N = \mathsf{F}\times\ldots\times\mathsf{F}$. In particular, a metric between two N-tuples of *FIN*s $A=(A_1,\ldots,A_N)$ and $B=(B_1,\ldots,B_N)$ can be computed using the following *Minkowski* metric

$$d_p(A,B)=[(d_K(A_1,B_1))^p+\ldots+(d_K(A_N,B_N))^p]^{\frac{1}{p}}, \text{ where}$$

$d_K: \mathsf{F}\times\mathsf{F}\rightarrow\mathsf{R}_0^+$ is the metric between *FIN*s shown in proposition 7.

3.4 Convexity in F

The obvious next step is to extend both addition and multiplication from M^h, $h\in(0,1]$ to F.

Definition 8

The product kF_1, where $k\in\mathsf{R}$ and $F_1\in\mathsf{F}$, is defined as *FIN* F_p: $F_p(h)=kF_1(h)$, $h\in(0,1]$.

■

We remark that the product kF_1 is always a *FIN*. More specifically for a positive nontrivial *FIN* F_1 in F_+, it follows that if $k\geq 0$ then $kF_1\in\mathsf{F}_+$, whereas if $k<0$ then $kF_1\in\mathsf{F}_-$; and vice-versa for a negative *FIN* F_1 in F_- that is, if $k>0$ then $kF_1\in\mathsf{F}_-$, whereas if $k\leq 0$ then $kF_1\in\mathsf{F}_+$. Apparently for a trivial positive *FIN* F the product kF is always a trivial positive *FIN*. From

the previous analysis it follows that both sets of positive *FIN*s (F_+) and negative *FIN*s (F_-) are cones. Examples of products kF_1 are shown in Fig.3 where $F_p = -F_n$ moreover $F_m = -2F_q$.
Consider the following definition for the sum of two *FIN*s.

Definition 9

The sum F_1+F_2, where $F_1,F_2 \in \mathsf{F}$, is defined as F_s: $F_s(h)=(F_1+F_2)(h)= F_1(h)+F_2(h)$, $h \in (0,1]$.

■

We remark that when both F_1 and F_2 are in the cone F_+ (F_-) then F_1+F_2 is a *FIN* in the cone F_+ (F_-). However a problem may arise in calculating the sum F_1+F_2 involving one positive and one negative *FIN*. The aforementioned problem occurs when the generalized interval $F_1(h)+F_2(h)$ is both *positive* for some values of $h \in (0,1]$ and *negative* for other values of $h \in (0,1]$. In the latter case, the sum F_1+F_2 is not a *FIN*.
Of particular interest in the context of this work are convex combinations $kF_1+(1-k)F_2$, $k \in [0,1]$, where F_1 and F_2 both are positive *FIN*s. Fig.5 shows two positive *FIN*s, respectively, *A* and *B*; moreover, the linear convex combination $kA+(1-k)B$ is shown for various values of *k*, i.e. $k=0.8$, $k=0.6$, $k=0.4$ and $k=0.2$. Fig.5 demonstrates that the positive *FIN* '$kA+(1-k)B$' is progressively a combination of both the location and the shape of positive *FIN*s *A* and *B*.

4. Construction and interpretations for a *FIN*

A *FIN* in this work is induced from a population of real number samples using algorithm CALFIN described elsewhere (Kaburlasos, 2004; Kaburlasos & Papadakis, 2004; Petridis & Kaburlasos, 2003). More specifically, algorithm CALFIN constructs a positive *FIN* with membership function $\mu(x)$ such that $\mu(x)=1$ for exactly one number *x*. By construction, algorithm CALFIN implies a one-to-one correspondence between *FIN*s and (*probabilistic*) *cumulative distribution functions* (*CDF*s) as explained in the following. In the one direction, a *CDF* $G(x)$ is mapped to a *FIN F*: Let x_0 be such that $G(x_0)=0.5$. The membership function $\mu_F(.)$ of the corresponding *FIN F* is defined such that $\mu_F(x)=2G(x)$ for $x \le x_0$ furthermore $\mu_F(x)=2[1-G(x)]$ for $x \ge x_0$. In the other direction, a *FIN F* is mapped to a *CDF*: Let x_0 be such that $\mu_F(x_0)=1$. The corresponding *CDF* $G(x)$ is defined such that $G(x)=0.5\mu_F(x)$ for $x \le x_0$ furthermore $G(x)=1-0.5\mu_F(x)$ for $x \ge x_0$. Note that, in general, the membership function $\mu_F(.)$ of a *FIN F* is nonparametric.
When algorithm CALFIN is applied on a population $x_1,x_2,\ldots,x_n$ of samples and a *FIN F* is constructed then $100(1-h)\%$ of the *n* samples $x_1,x_2,\ldots,x_n$ are within interval $\{x: F(h)\neq 0\}$; the remaining $100h\%$ of the samples $x_1,x_2,\ldots,x_n$ are split equally both below and above interval $\{x: F(h)\neq 0\}$. If a large number of samples is drawn independently according to a probabilistic

distribution function $p_0(x)$ and a *FIN* F is constructed using algorithm CALFIN then the interval $\{x: F(h)\neq 0\}$ constitutes, by definition, an *interval of confidence at level-h* in the following sense: A random number drawn according to $p_0(x)$ is expected to fall 1) *inside* interval $\{x: F(h)\neq 0\}$ with probability 100(1-h)%, 2) *below* interval $\{x: F(h)\neq 0\}$ with probability 50h%, and 3) *above* interval $\{x: F(h)\neq 0\}$ with probability 50h%. Due to the aforementioned "one-to-one correspondence" between *FIN*s and *CDF*s it follows that a *FIN* captures statistics of all orders. The computation of *FIN*s is demonstrated in the following.

Fig.6 displays *FIN*s constructed by algorithm CALFIN from data samples generated according to the *normal* probability density function N(0,1) with mean 0 and standard deviation 1. In particular, 50 random data samples were used to compute *FIN* G1 in Fig.6(a), whereas 10,000 random data samples were used to compute *FIN* G2 in Fig.6(b). Note that the maximum value, i.e. one, for either *FIN* G1 or G2 is attained as expected near the *mean* 0 of the probability density function N(0,1). A comparison of figures Fig.6(a) and Fig.6(b) confirms that, as the number of samples increases, the corresponding *FIN* converges asymptotically to a limit. When the number of samples tends to infinity then the computed *FIN* corresponds to the Gaussian *CDF*. Note that both the left and the right tails of *FIN* G2 in Fig.6(b) asymptotically tend to zero due to the nature of the Gaussian *CDF*.

Another example is shown in Fig.7, where *FIN*s are displayed constructed by algorithm CALFIN from data samples drawn according to a *uniform* probability density function over the range [-3,3]. In particular, 50 random data samples were used to compute *FIN* U1 in Fig.7(a), whereas 10,000 random data samples were used to compute *FIN* U2 in Fig.7(b). A comparison of figures Fig.7(a) and Fig.7(b) confirms that as the number of samples increases then the computed *FIN* converges asymptotically to its limit. We remark that *FIN* U2 in Fig.7(b) has an isosceles triangular membership function as expected for a uniform probability density.

A positive *FIN* constructed by algorithm CALFIN has been interpreted (above) "statistically" as an estimate of a (probabilistic) cumulative distribution function. In a different context, using a fuzzy set theoretic terminology, a positive *FIN* F may be interpreted as a nonparametric *possibility distribution* (Zadeh, 1978), where the corresponding fuzzy membership function $\mu_F(.)$ is perceived as a (linguistic) constraint (Zadeh, 1999). A statistical interpretation for a *FIN* does not exclude a linguistic (fuzzy) interpretation. For instance, Dubois & Prade (1986) have presented transformations between probability and possibility measures; moreover Dubois & Prade (1986) present algorithms, similar in spirit with algorithm CALFIN, for inducing fuzzy membership functions from statistical interval data.

Under any interpretation, a *FIN* is an *information granule* that is a clump of values drawn together by indistinguishability, similarity, proximity or functionality (Zadeh, 1999; Bortolan & Pedrycz, 2002; Pedrycz, 2002; Pedrycz & Bargiela, 2002).

Note that a trivial *FIN* $\bigcup_{h\in(0,1]}[x,x]^h$ represents a crisp real number x. Moreover, a positive *FIN* $\bigcup_{h\in(0,1]}[a,b]^h$, $a,b\in\mathsf{R}$ with $a\leq b$ represents a conventional crisp interval $[a,b]$. A fuzzy membership function has been proposed in the literature as follows $u(x)=1/(1+d(x,x_0))$ (Krishnapuram & Keller, 1993), where $d(x,x_0)$ is the distance of a point x from a class prototype point x_0. Likewise, fuzzy membership function $\mu_F(H)= 1/(1+d_K(F,H))$ could be used in a future work, where F and/or H are *FIN*s including fuzzy sets, intervals, and real numbers.

5. Granular self-organizing map (*grSOM*)

Kohonen's self-organizing map (KSOM) typically implements a nonlinear projection of a probability density function $p(x)$ from a high dimensional input data space R^N onto a two-dimensional grid of units/neurons such that topology is preserved. The weights of a neuron in the grid are updated by a linear convex combination

$$m_i(t+1) = m_i(t) + h(t)[x(t)-m_i(t)] = [1-h(t)]m_i(t) + h(t)x(t), \quad 0<h(t)<1,$$

where $m_i(t)\in\mathsf{R}^N$ is a reference (weight) vector, $x(t)\in\mathsf{R}^N$ is an input (stimulation) vector, and $h(t)$ is the *neighborhood function*.

A number of KSOM variants have been presented in the literature, typically in signal processing applications (Kohonen, 1995). A different KSOM variant is presented below towards linguistic (fuzzy) system modeling applications.

5.1 Principles for extending KSOM

An extension of Kohonen's self-organizing map (KSOM) is feasible, as explained below, in the metric space F^N. A motivation for the aforementioned extension is that positive *FIN*s can accommodate ambiguity. In view of the material presented in sections 3 and 4, an extension of KSOM from R^N to F^N is straightforward based on: 1) algorithm CALFIN, for computing a *FIN* from a population of real number samples in a data dimension, 2) a Minkowski metric in F^N, and 3) convex combinations of *FIN*s. An extended KSOM algorithm in F^N is called *granular SOM* or *grSOM* for short, whereas KSOM will also be called here *crisp SOM*. A simple *grSOM* algorithm, namely *greedy grSOM*, is presented in the following.

5.2 The 'greedy grSOM' algorithm

The *training phase* of the *greedy grSOM* algorithm is shown in Fig.8.

The clustering carried out by the *greedy grSOM* is supervised in the sense that to each unit 'ij', i=1,…,I, j=1,…,J in the grid the algorithm ultimately assigns the label of the category which provided the majority of the training data during all epochs for giving rise to the weight

$W_{ij} \in \mathsf{F}^N$ (step-5 in Fig.8). We point out that during training (steps 3 and 4 in Fig.8) the labels are not used. A grid unit with both weight $W_{ij} \in \mathsf{F}^N$ and label L_{ij} is interpreted as a fuzzy rule "if W_{ij} then L_{ij}". Therefore, the *greedy grSOM* can be regarded as a fuzzy inference system (FIS); more specifically the *greedy grSOM* is a fuzzy neural network for classification.

The task of the *greedy grSOM* is to track down clusters in the training data and put a *FIN* on a cluster so as to "fully cover" the training data. By "full coverage" we mean that the interval supports of all *FIN*s, involved in the antecedent (IF) parts of fuzzy rules, cover the training data. Note that if, as expected in practice, the training data domain equals the testing data domain then "full coverage" is important because it guarantees that at least one fuzzy rule will be activated for an input. That is why the *grSOM* algorithm in Fig.8 is named *greedy*: because it expands its fuzzy rule supports so as to cover the training data.

The *greedy grSOM* algorithm (Fig.8) employs both algorithm CALFIN and a Minkowski metric in F^N, nevertheless it does not employ convex combinations of *FIN*s. There follow advantages and disadvantages. More specifically, a disadvantage is that the *greedy grSOM* needs to "batch process" the whole training data set in order to compute a new weight value $W'_{ij} \in \mathsf{F}^N$. We plan to develop in the future an *incremental grSOM* algorithm using convex combinations of *FIN*s. However, the latter algorithm may leave part of the training data outside all fuzzy rule interval supports (for a better understanding consult Fig.5), whereas the *greedy grSOM* guarantees full coverage of the training data domain; the latter is an advantage in linguistic FIS applications as explained above. Furthermore note that, on the one hand, the *crisp SOM* computes N-dimensional reference (weight) vectors $m_i(t)$ using convex combinations in R^N; hence *KSOM* captures, locally, only first-order statistics in the training data. On the other hand, the *greedy grSOM* here computes a distribution of *FIN*s. Based on the one-to-one correspondence between *FIN*s and probabilistic distribution functions, it follows that the *greedy grSOM* captures locally all-order statistics in the training data. Comparisons of the *greedy grSOM* with alternative *KSOM* variants are presented below.

Initialization of unit weights in the grid is effected by assigning, randomly, training data to grid units (Fig.8). The training phase proceeds by assigning an input datum to the winner of the competition among the units in the grid, where the criterion for winning is (metric) proximity of an input N-tuple of *FIN*s in F^N to a grid unit weight $W_{ij} \in \mathsf{F}^N$, i=1,…,I, j=1,…,J. We point out that an input datum in F^N may be nontrivial.

In *crisp SOM* the winner's neighbor units are trained less than the winner unit according to the shape of the, typically bell-shaped, neighborhood function. The version of *greedy grSOM* presented in Fig.8 employs, for simplicity, a crisp neighborhood; in other words, a unit either belongs or does not belong to the winner unit's neighborhood. Therefore, the *greedy grSOM* here trains the winner unit's neighbors as much as it does the winner unit itself.

Initially the neighborhood size $N_{pq}(t)$ of a winner unit 'pq' is chosen large enough to enclose about half of the grid. Progressively, after a number of epochs, the neighborhood size of the winner unit decreases and, eventually, an input datum is assigned only to the winner unit 'pq'. No learning takes place for *greedy grSOM* before the end of an *epoch*, which occurs when all the input data have been presented. Then a new value W'_{ij} is computed (step-4 in Fig.8) for a unit weight $W_{ij} \in \mathsf{F}^N$, i=1,…,I, j=1,…,J from all the input data assigned to grid unit 'ij' during the current epoch. The terminating condition for the *greedy grSOM* is a user-defined total number *Nepochs* of epochs. As soon as training completes, labels are assigned to grid nodes.

Similar to *KSOM*, the *greedy grSOM* algorithm cannot guarantee that each unit in the grid will be assigned to a category. In fact the experiments below show that some units may stay unused. We point out that, apart from SOM variants, other clustering algorithms may result in unused units, e.g. the *k-means* algorithm (Duda, Hart, & Stork, 2001).

A trained *greedy grSOM* can be used for testing. In particular, an input datum X is assigned to the category whose label L_{pq} is attached to the nearest grid unit weight W_{pq}, i.e. W_{pq}= $\underset{\substack{i=1,\ldots,I \\ j=1,\ldots,J}}{arg\ min}\ d_1(X,W_{ij})$, where $d_1(.,.)$ is a Minkowski metric between two N-tuple *FIN*s in F^N.

The *greedy grSOM* algorithm employs the metric distance function $d_K(.,.)$ in the space F of *FIN*s, where a *FIN* in this work emerges from a population of real number samples. There are alternative distance functions between either fuzzy numbers or populations of samples including 1) the Hausdorf metric d_H, and 2) the Kullback-Leibler distance d_{KL}. The former distance (d_H) is a *metric* distance typically employed in mathematically oriented publications (Körner & Näther, 2002); nevertheless d_H may produce non-sensible results in practical applications (Kaburlasos, 2004). The latter distance (d_{KL}) is a sensible distance function; however d_{KL} is not a metric (Kaburlasos, 2004); this may be a problem for rigorous design. It turns out that distance d_K is both *metric* and *sensible* (Kaburlasos, 2004); therefore here we employed d_K. Note also that, on the one hand, the *KSOM* typically uses either the Euclidean distance or the city-block distance between N-dimensional points (Kohonen, 1995). On the other hand, a *grSOM* algorithm can use a more general Minkowski metric distance $d_p(.,.)$, p=1,2,…. In the interest of simplicity the Minkowski metric $d_p(.,.)$ for p=1 was used in this work. Even restricted in $d_1(.,.)$ there is an infinite number of metric distances for the *greedy grSOM* as explained in the following.

5.3 Genetic algorithm (GA) optimization

The *greedy grSOM* algorithm employs the Minkowski metric d_1: $\mathsf{F}^N \times \mathsf{F}^N \to \mathsf{R}_0^+$. Chapter 3 has shown that the definition of d_1 is based on the metric d_K: $\mathsf{F} \times \mathsf{F} \to \mathsf{R}_0^+$; in turn, the definition of d_K is based on metric d_h: $\mathsf{M}^h \times \mathsf{M}^h \to \mathsf{R}_0^+$; finally, the definition of the latter is based on a *strictly increasing* function f_h: $\mathsf{R} \to \mathsf{R}$. An improvement of classification performance for the *greedy grSOM* was pursued here by computing optimally a *strictly increasing* function in each data dimension using a genetic algorithm (GA) as explained in this section.

Here we avoided the time consuming task of computing a *strictly increasing* function from a mass function; therefore, we considered the following (parametric) *strictly increasing* function in each data dimension in R^N:

$$f_i(x) = c_{i,1}\tanh\left(\frac{x - a_{i,1}}{b_{i,1}}\right) + c_{i,2}\tanh\left(\frac{x - a_{i,2}}{b_{i,2}}\right) + c_{i,3}\tanh\left(\frac{x - a_{i,3}}{b_{i,3}}\right), \quad i=1,\ldots,N$$

where tanh(.) is the 'hyperbolic tangent' function, $a_{i,j}, b_{i,j}, c_{i,j} \in \mathsf{R}$, i=1,…,N, j=1,2,3. Illustrations regarding the aforementioned functions are provided in the following.

Fig.9(a) plots the saturated (step) function tanh(x), whereas Fig.9(b) plots function $f(x) = c_1(\tanh(x-a_1)/b_1) + c_2(\tanh(x-a_2)/b_2) + c_3(\tanh(x-a_3)/b_3)$ for selected values of the parameters a_i, b_i, c_i, i= 1,2,3. Note that the latter function $f(x)$ is the superposition of three 'component' saturated functions tanh(.), where the location, scale and height of each component function are specified, respectively, by the corresponding parameters a_i, b_i, and c_i, i=1,2,3. The reason for using more than one component functions tanh(x) in f(x) above is for introducing "tunable flexibility" in applications. In all, a function $f(x)$, i=1,…,N includes 9 parameters a_i, b_i, c_i, i= 1,2,3, hence a total of 9N parameters are required to define $f_i(x)$, i=1,…,N functions in R^N.

Given a classification problem in R^N the objective is to calculate optimal estimates for the strictly increasing functions $f_i(x)$, i=1,…,N. The aforementioned optimization was pursued using a genetic algorithm (GA) immediately after training the *greedy grSOM*. Note that GAs is a popular optimization technique in both neural and fuzzy modeling applications (Chun & Chang, 2000; Ishigami *et al*, 1995; Shimojima, Fukuda, & Hasegawa, 1995). An individual solution of the GA for the *greedy grSOM* was represented here using 9N parameters. Therefore the chromosome of an individual solution consisted of 9N genes, and each gene used 16 bits to encode a single parameter value. In conclusion, a chromosome was 144N bits long. The population of the genetic algorithm involved 25 individuals. The genetic algorithm employed multipoint crossover and roulette wheel selection for the reproduction process, elitism, multipoint mutation, and adaptive crossover-mutation rates. The genetic optimization scheme was enhanced using specialized operators such as the Adaptive Search space Range (ASER) (Papadakis & Theocharis, 1996), and the microgenetic algorithm (Kazarlis *et al*,

2001). The classification performance on the training data set was used as the fitness value of an individual. The evolution terminated when the fitness of the elite individual was not improved for 20 generations in a row.

We now compute the time complexity of the *greedy grSOM* (Fig. 8). The *greedy grSOM* includes an external loop of *Nepochs* epochs. In each epoch certain operations are carried out on the I×J grid in each one of the N data dimensions. In particular, 1) *FIN*s are computed from n data, and 2) the metric $d_K(.,.)$ is computed between two *FIN*s. On the one hand, the divide-and-conquer algorithm CALFIN has O($n$log$n$) computational time complexity with respect to the number 'n' of data to compute a *FIN* in a data dimension. On the other hand, the computation of metric $d_K(.,.)$ requires a summation of L terms for computing the integral in proposition 7 for height h values from 0 to 1. In conclusion, the computational complexity of the *greedy grSOM* algorithm for training equals O(*Nepochs*IJNn(L+log$n$)). As soon as *greedy grSOM* training terminates the GA algorithm for optimization is launched. The GA carries out a stochastic search and it requires substantial additional time for clearly improving classification performance as explained in the experiments section below.

6. The *greedy grSOM* in perspective

This section relates the *greedy grSOM* to various models from the literature.

6.1 Neural models based on lattice theory

The techniques presented here build on a growing body of work regarding an improvement of neural computing models based on lattice theory. Previous work has presented the *σ-FLN* neural network as an extension of Carpenter's fuzzy version of Grossberg's adaptive resonance theory (ART) to a lattice data domain (Kaburlasos & Petridis, 2000). Several fuzzy lattice neurocomputing (FLN) models have been introduced for clustering and classification. More specifically the *σ-FLN*, *σ-FLNMAP*, *dσ-FLN*, and *FLNtf* models have demonstrated a capacity to process, either separately or jointly, disparate types of data including vectors of numbers, (fuzzy) sets, symbols, linear operators (matrices), hyperspheres, Boolean propositions, waveforms, and graphs (Kaburlasos *et al*, 1999; Kaburlasos & Petridis, 2000, 2002; Petridis & Kaburlasos, 1998, 2001).

Apart from the effectiveness of FLN models in disparate data domains, our previous work has elevated lattice theory as a tool for analysis and design. For instance, an instrument typically employed by a FLN model is an *inclusion measure* function $\sigma: L\times L\rightarrow[0,1]$, which specifies a fuzzy degree of inclusion of one lattice L element into another one. It has been shown that a real *positive valuation* function v: $L\rightarrow R$ can define an inclusion measure σ in a lattice

(Kaburlasos & Petridis, 2000). In addition a positive valuation introduces a metric distance function d: $\mathsf{L}\times\mathsf{L}\rightarrow\mathsf{R}_0^+$, which can be used for extending the applicability of KSOM as explained in the following.

6.2 Crisp and fuzzy KSOM extensions

Several authors have proposed KSOM extensions to nonnumeric data including symbol strings (Kohonen, 1996; Kohonen & Somervuo, 1998; Somervuo, 2004), qualitative (nominal) data (Cottrell, Ibbou, & Letrémy, 2004), and matrices (Seo & Obermayer, 2004) based on domain-specific distance functions. Furthermore, it is interesting to point out that various authors have sought KSOM extensions using nonEuclidean (Riemannian) metrics (Peltonen, Klami, & Kaski, 2004; Ritter, 1999). A motivation for considering lattice extensions of KSOM is that lattice theory can supply metric distances based on tunable positive valuation functions. For instance the metric $d_K(.,.)$, presented above in the set F of fuzzy interval numbers, can be used for extending KSOM to a linguistic (fuzzy) data domain.

Crisp clustering algorithms, including KSOM, cannot cope with ambiguity in applications (Krishnapuram & Keller, 1993). Additional problems for KSOM include the lack of sound optimization and convergence strategies (Tsao, Bezdek, & Pal, 1994). In response, a number of "Kohonen type" fuzzy *c*-means algorithms have been proposed (Karayiannis & Bezdek, 1997; Tsao, Bezdek, & Pal, 1994; Pal, Bezdek, & Tsao, 1993). Note that the aforementioned fuzzy algorithms process crisp data, more specifically they process vectors of numbers. Alternative fuzzy KSOM extensions have been proposed for processing linguistic (fuzzy) data using simplified 3-dimensional vector representations for the linguistic data (Mitra & Pal, 1994, 1996). The latter KSOM extensions can be interpreted as simple fuzzy inference systems (FIS) whose rule consequents are category labels. Note that none of the aforementioned fuzzy KSOM extensions pursues explicitly a statistical data interpretation. Moreover, all the aforementioned extensions employ (implicitly) the constant mass function $m(x)=1$. It turns out that the above algorithms use parametric *possibility distributions* (i.e. fuzzy sets) as devices for introducing tunable nonlinearities in pattern classification problems.

A different fuzzy extension of KSOM has been reported for implementing a fuzzy inference system (FIS) in a function f: $\mathsf{R}^N\rightarrow\mathsf{R}$ approximation problem (Vuorimaa, 1994). In particular, the centers of fuzzy sets are learned using the *crisp SOM*. Then, fuzzy sets with triangular membership functions are inserted and fine-tuned. A grid node defines a fuzzy rule with a singleton number consequent. Finally, a weighted average of activated rule outputs produces a continuous valued output. The work in Vuorimaa (1994) is the nearest work we are familiar with in the literature towards a fuzzy inference systems (FIS) extension of KSOM. However, the work in Vuorimaa (1994) considers solely triangular fuzzy membership functions, the

inputs to a fuzzy SOM cannot be fuzzy and, finally, the constant mass function $m(t)=1$ is employed implicitly; moreover the importance of structure identification is not recognized.

There is an inherent relation between fuzzy set theory and lattice theory based on the lattice ordering of the fuzzy sets (Zadeh, 1965). Furthermore, there is an inherent relation between probability theory and lattice theory based, likewise, on the lattice ordering of events in a probability space (Birkhoff, 1967). Hence, lattice theory might be a sound mathematical foundation for unifying probability theory and fuzzy set theory. Note that various authors have proposed theoretical connections between fuzzy set theory and probability theory (Goodman & Nguyen, 2002; Körner & Näther, 2002). Related work in practical applications is described in the following.

6.3 Links with probabilistic and statistical models

A synergy of fuzzy modeling techniques with statistical techniques has been proposed towards an optimization of fuzzy model construction in a principled manner (Yen & Wang, 1998). The latter work retains conventional fuzzy set interpretations, moreover statistical information criteria are proposed as effective cost functions in nonlinear function approximation problems. In a different context, more specifically in the area of pattern recognition, a connection has been shown between fuzzy sets and statistics (Kuncheva, 1996); more specifically, a theoretical equivalence was shown between fuzzy systems and two nonparametric statistical classifiers from both a functional and a morphological perspective. On the other hand, the work here proposes an extension of KSOM for classification, which induces a distribution of fuzzy interval numbers (*FIN*s) from the data. Since a *FIN* may represent a local probabilistic distribution function, it is natural to compare *grSOM* directly with various probabilistic mixture models in the following.

A KSOM extension for clustering and nonlinear dimensionality reduction is described in Verbeek (2005) based on mixture densities and *greedy* expectation-maximization (EM) parameter estimation algorithms; the aforementioned term ‘greedy’ refers to an increasing number of components in the mixture. A similar algorithm, namely self-organizing mixture model (SOMM) applicable as well on discrete and/or continuous data, has been reported elsewhere (Verbeek, Vlassis, & Kröse, 2005). Self-organizing behavior by SOMM is obtained by constraining (i.e. normalizing) the distribution q_n on the mixture components for a data point x_n. Advantages of SOMM include an optimization of a well-defined objective function, and a principled handling of missing data values based on probability theory. However, a SOMM does not consider alternative divergence (≡ distance) functions. Furthermore a SOMM is applicable solely on crisp data and it cannot cope with linguistic data.

Another KSOM extension based on mixture models is the generative topographic mapping (GTM) (Bishop, Svensén, & Williams, 1998). A comparison of GTM with the

abovementioned SOMM is shown in Verbeek, Vlassis, & Kröse (2005). The GTM is an extension of the latent (hidden) variable framework to allow nonlinear transformations. The operation of GTM is based on a constrained mixture of Gaussians whose parameters can be optimized by maximum likelihood using the expectation-maximization (EM) algorithm. The GTM provides a principled alternative to KSOM, and overcomes significant limitations of KSOM. For instance, the GTM defines explicitly an objective function given by a log likelihood; furthermore, convergence to a (local) maximum of the objective function is guaranteed by the use of the EM algorithm. An important application of GTM is to data visualization. There are substantial differences between a probabilistic mixture model and a *grSOM* model as explained next.

A mixture model pursues analytically an optimization of a parametric objective function. The inspiration for an objective function typically derives from energy functions in physics. Nevertheless energy functions cannot accommodate linguistic data such as fuzzy sets and conventional intervals. The latter data can handle ambiguity in technological applications. Furthermore, a mixture model typically assumes (a priori) parametric probability density functions, e.g. Gaussian functions, etc. In the context of this work fuzzy interval numbers (*FIN*s) are employed for representing nonparametric probability density functions. Moreover, parametric mass functions are employed here for introducing tunable nonlinearities. There is another substantial difference between a *grSOM* model and a mixture model. On the one hand, a mixture model seeks optimization analytically using (implicitly) mass function $m(x)=1$ in a data dimension. On the other hand, a *grSOM* model seeks optimization by genetic search looking for a different mass function in a data dimension. The objectives of a mixture model typically include dimensionality reduction and visualization, whereas the principal objective of a *grSOM* model is extraction of descriptive decision-making knowledge (fuzzy rules) from the training data. Finally, from a rigorous mathematical point of view a *grSOM* model treats the Euclidean space R^N differently than a mixture model. More specifically, the latter treats R^N as a real linear space; whereas a *grSOM* model treats R^N as the Cartesian product of N totally-ordered lattices R. Hence a *grSOM* model can adhere to linguistic semantics; the latter is especially sensible when different quantities are involved in different data dimensions, e.g. weight, speed, etc.

6.4 Novelties in this work

Part of the material of this work has been published elsewhere, mainly in Kaburlasos (2004) and in Kaburlasos & Papadakis (2004). This work includes significant enhancements as well as substantial novelties as summarized in this section.

The operation of the *greedy* grSOM is heuristic like Kohonen's SOM; however, the *greedy* grSOM retains linguistic interpretations. Even though no objective function is employed by

the *greedy grSOM*, nevertheless this work has laid a sound mathematical foundation towards further future improvements. For instance, based on convex combinations of *FIN*s, a *grSOM* model could extend the *crisp SOM* or the *k*-means algorithm (Duda, Hart, & Stork, 2001) to the metric convex set $\mathbb{F}$ of *FIN*s. In this context objective functions could be devised involving, as well, linguistic data. Note also that the treatment of convexity in $\mathbb{F}$ is a novelty of this work.

Previous work has introduced *mass functions*. The effectiveness of mass functions is demonstrated here in practice for the first time. More specifically, mass functions are employed here as an instrument for improving classification performance by introducing tunable nonlinearities. It is interesting that other authors have proposed using different weighting factors $\lambda_i \geq 0$, i=1,…,n for different input data dimensions in order to improve the classification performance of a KSOM variant (Hammer & Villmann, 2002). Using the terminology of this work, a weight factor corresponds to mass function $m_i(t)= \lambda_i$. Hence, a factor λ_i attaches the same weight of significance to all numbers along a single data dimension; whereas, a mass function $m_i(t)$ may attach, more flexibly, different weights of significance to different numbers along a single data dimension.

Another major novelty of this work concerns the employment of genetic algorithms (GA) for optimizing classification performance. Note that various authors have pursued a GA optimization of Self-Organizing Maps. For a fairly recent overview the reader may refer to Polani (1999), where a new GA is also proposed for improving a degree of SOM organization while preserving topology. Note in addition that the performance of fuzzy control systems can be improved genetically by tuning parameterized membership functions as well as input-output scaling factors (Hoffmann, 2001). The important difference here is that a GA computes optimally mass functions for tuning a metric distance between nonparametric fuzzy interval numbers (*FIN*s).

Additional novelties include an improved notation in a unified mathematical presentation, an analysis of the complexity of the *greedy grSOM* algorithm, extensive comparisons with related work from the literature, new experiments with improved classification results, and a display of linguistic (fuzzy) rules induced from the data as shown in the following section.

7. Experimental Results

We considered three benchmark classification problems involving two or three classes in a problem. The data sets were downloaded from the UCI machine-learning repository (Blake & Merz, 1998). In a preprocessing step each data attribute was normalized in the range [0,1] by a linear transformation, which mapped the minimum and maximum attribute values,

respectively, to the numbers 0 and 1. Computational experiments were carried out with SOM variants on a 4×4 grid of units. A SOM variant was trained for *Nepochs*= 10,000 epochs.

Ten different random permutations of the data in a class were considered. The first part of the data in a random permutation was left out for testing, whereas the remaining data were employed for training. Typically two different data partitions were considered: 1) 90% of the data for training and 10% for testing, and 2) 66% of the data for training and 34% for testing. Care was taken so that all data classes were represented balanced in both the training- and the testing data. In conclusion, ten different pairs of training-testing data sets have been available for each data partition.

Training experiments were carried out on the same data sets using, first, the *crisp SOM* and, second, the *greedy SOM* with mass function $m(x)=1$. As soon as *greedy SOM* training completed, the training resumed towards an optimization of the mass functions using a genetic algorithm (GA). All SOM variants here employed crisp neighborhoods. In every experiment the testing data were used only once for a SOM algorithm.

7.1 The FISHER IRIS benchmark

This is perhaps the best-known data set in the pattern recognition literature. It contains 4-dimensional vectors that correspond to sepal/petal length and width of flowers in three classes including 50 vectors per class. A training data set is not given explicitly. One flower class is linearly separable from the other two; the latter are not linearly separable from each other. The goal was to predict the correct flower class in the testing data. This problem is characterized by low misclassification rates.

Our results are summarized in Table 1, where for the two data partitions the classification accuracies of various SOM models are shown on both the training- and the testing data. More specifically, the average accuracies in ten computational learning experiments are shown as well as the corresponding standard deviations (stdv); in addition, the corresponding numbers of grid units are displayed in Table 1. Fig. 10 shows all the eight fuzzy rules induced by the *greedy grSOM* in one of the ten experiments. Fig. 11 shows the corresponding optimal mass functions computed on each of the four data dimensions. A detailed discussion of the results is presented below.

7.2 The WINE RECOGNITION benchmark

The WINE RECOGNITION benchmark data set includes 13-dimensional vectors that correspond to various wine constituents whose values are measured chemically. In all, there are 178 data vectors partitioned in three wine classes including 59, 71, and 48 data vectors, respectively. A training data set is not given explicitly. The goal was to predict the wine class in the testing data.

Table 2 summarizes the classification results by other algorithms. Results by Regularized Discriminant Analysis (RDA), Quadratic Discriminant Analysis (QDA), Linear Discriminant Analysis (LDA), and 1-Nearest-Neighbor (1NN) are reported in the UCI repository (Blake & Merz, 1998) using the leave-one-out technique. Results by the remaining algorithms in Table 2 are from Joshi *et al* (1997) where two-thirds of the data have been used for training and the remaining one-third for testing.

Our experimental results are summarized in Table 3, where for two data partitions the classification accuracies of various SOM models as well as the corresponding standard deviations (stdv) are shown on both the training- and the testing data; in addition, the corresponding numbers of grid units are displayed in Table 3.

7.3 The CLEVELAND HEART DISEASE benchmark

The CLEVELAND HEART DISEASE benchmark data set, authored by R. Detrano, in its functional version contains 14-dimensional vectors, which include both vital signs and heart disease attributes. In all, there are 164 and 139 data vectors, respectively, from classes 0 (absence of heart disease) and 1 (presence of heart disease). There are a few missing data attributes, which have been replaced here by the average value of the corresponding attribute. A training data set is not given explicitly. The goal was to predict absence/presence of heart disease in the testing data.

Table 4 summarizes the classification results by different algorithms. Results by Logistic Regression, Conceptual Clustering (CLASSIT), Discriminant Analysis, Instance Based Prediction (both NTgrowth and C4) are reported in the UCI repository (Blake & Merz, 1998). Results by the ARTMAP-IC, Fuzzy ARTMAP, and kNN are from Carpenter and Markuzon (1998), where all ARTMAP results reflect the participation of ten voters. Results by FLNtf are from Kaburlasos & Petridis (2000). Apart from the 10-fold cross-validation of FLNtf, all the other algorithms in Table 4 have employed 250 data vectors for training and the remaining 53 data vectors for training.

Our experimental results are summarized in Table 5, where for two data partitions the classification accuracies of various SOM models as well as the corresponding standard deviations (stdv) are shown on both the training- and the testing data; in addition, the corresponding numbers of grid units are displayed in Table 5.

7.4 Discussion of the results

Our ten different experiments in a data partition have demonstrated variability as expected. Therefore a test of significance was necessary. For this reason Tables 1, 3, and 5 show as well the corresponding standard deviation (stdv) values. In addition, we carried out standard "statistical hypothesis testing" as described next.

Each data set was processed using three SOM variants: 1) the *crisp SOM*, 2) the *greedy grSOM* (with $m(x)=1$), and 3) the *GA optimized greedy grSOM*. After performing ten experiments for a data partition, the algorithms were evaluated pair wise using the one-sided "matched pairs" statistical t test with df=9 degrees of freedom. The null hypothesis H_0: "the two algorithms in a pair give similar results" was tested versus the alternative hypothesis H_a: "the second algorithm in a pair improves classification performance". In each case we computed the P-value of the corresponding statistic $t= (\bar{x}-0)/(s/\sqrt{n})$ for $n=10$, where $\bar{x}$ is the sample average of differences in classification accuracy and s is the corresponding standard deviation. We worked at 5% level of significance.

For the IRIS 90%-10% data partition, a comparison of testing data accuracy of the *crisp SOM* (average 93.33%) with the *greedy grSOM* (average 96.66%) resulted in $t=2.23$ which implied $P=0.0261$; whereas a testing performance comparison of the *crisp SOM* (average 93.33%) with the *GA optimized greedy grSOM* (average 98.70%) resulted in $t=2.75$ which implied $P=0.0112$. Hence the null hypothesis H_0 could not be accepted in any of the above statistical tests; in other words, a *greedy grSOM* algorithm appears to improve the testing classification accuracy. Nevertheless, a testing performance comparison of the *greedy grSOM* (average 96.66%) with the *GA optimized greedy grSOM* (average 98.70%) resulted in $t=1.15$ which implied $P=0.1394$. Therefore the null hypothesis H_0 could not be rejected; in other words, the mass functions do not appear to improve the testing classification accuracy in this case.

Table 1 shows that training data accuracy sometimes is smaller than the corresponding testing data accuracy for a SOM algorithm. It was confirmed that the difference in classification accuracy between the training data and the testing data for a SOM algorithm in Table 1 is not statistically significant. We believe that the aforementioned difference is due to the very small number (15) of data available for testing in the 90%-10% data partition. More specifically, it is well known that 2 or 3 data in the IRIS benchmark are difficult to classify correct. Due to the very small size of the testing data set, it is most likely that the latter data are included in the training data set resulting in an inferior training data classification accuracy. The aforementioned differences were reversed in the 66%-34% data partition (see in the lower part of Table 1).

Further statistical testing for the IRIS 90%-10% data partition confirmed that the training data accuracy of the *greedy grSOM* (average 95.55%) versus the *crisp SOM* (average 92.88%) is statistically significant with $P=0.0415$; moreover the training data accuracy of the *GA optimized greedy grSOM* (average 97.77%) versus the *greedy grSOM* (average 95.55%) is statistically significant with $P=0.000131$. Hence it appears that the *greedy grSOM* learns better than the *crisp SOM*; moreover mass functions improve learning further.

Regarding the number of units recall that the *greedy grSOM* and the *GA optimized greedy grSOM* induce the same number of units by construction. A statistical comparison of the number of units of the *crisp SOM* (average 16.0) with the *greedy grSOM* (average 10.1) resulted in $P \cong 0.0$; hence, the *greedy grSOM* engages fewer grid units than the *crisp SOM*. Note that the different numbers of units used in different experiments by the *greedy grSOM* are due to the different weight W_{ij} initializations (see in Fig.8, step-2).

Similar statistical testing results were obtained for the IRIS 66%-34% data partition. Moreover note that a testing performance comparison of the *greedy grSOM* (average 95.60%) with the *GA optimized greedy grSOM* (average 97.60%) resulted in $t=3.35$ which implied $P=0.0042$. Therefore the null hypothesis H_0 could not be accepted; in other words, the mass functions here improve the testing classification accuracy.

Fig.10 shows eight rules R1,…,R8 induced in one of the ten experiments regarding the IRIS 90%-10% data partition. The antecedent (IF part) of a rule is the conjunction of four fuzzy statements shown (in a line) in four frames, respectively; moreover, the consequent (THEN part) of a rule is a class label. Note that one, three, and four rules correspond, respectively, to classes 1, 2, and 3. Recall that class 1 is linearly separable from the other two, therefore only one rule suffices for class 1; however, more rules are required to learn the nonlinearly separable classes 2 and 3. The corresponding mass functions $m_1(x),\ldots,m_4(x)$ in the four data dimensions $x_1,\ldots,x_4$ are shown in Fig.11, where a mass function was calculated as the derivative of a strictly increasing function in the corresponding data dimension. Fig.11 indicates that a mass function in a data dimension is (typically) unimodal. Mass function $m_2(x)$ obtains the most significant values peaking up to over 9 for normalized variable values around $x \approx 0.45$. Significant values also obtains mass function $m_3(x)$ which peaks up to over 4 for normalized variable values around $x \approx 0.25$. The other two mass functions $m_1(x)$ and $m_4(x)$ in this problem are fairly "flat" with smaller peak values. The mass functions in Fig.11 indicate not only a different significance put on different data dimensions but also a different significance put on different real numbers in the same data dimension.

For the WINE classification problem we recorded similar statistical testing results. In particular for the 90%-10% data partition, a testing performance comparison of the *greedy grSOM* (average 98.33%) with the *GA optimized greedy grSOM* (average 99.40%) resulted in $t=1.5$ which implied $P=0.0839$. Hence, mass functions do not appear to improve the testing classification accuracy in this case due to the small size of the corresponding testing data set likewise as with the IRIS problem above. Overall, the *greedy grSOM* produced better results than the *crisp SOM*; moreover, the *GA optimized greedy grSOM* further improved performance. A statistical comparison of the number of engaged grid units of the *crisp SOM*

(average 16.0) with the *greedy grSOM* (average 10.0) resulted in $P \cong 0.0$; hence, the *greedy grSOM* appears to engage fewer units in the grid than the *crisp SOM*.

For the HEART benchmark classification problem we recorded similar statistical testing results. More specifically note that for the 90%-10% data partition, a testing performance comparison of the *crisp SOM* (average 78.76%) with both the *greedy grSOM* (average 81.99%) and the *GA optimized greedy grSOM* (average 84.70%) implied $P=0.1233$ and $P=0.0703$, respectively. Hence, a *greedy grSOM* variant does not seem to improve the testing classification accuracy in this case. Nevertheless, for the 250-53 data partition the testing classification accuracy of both *greedy grSOM* variants was clearly better than the accuracy of *crisp SOM*; moreover, the performance of the *GA optimized greedy grSOM* was clearly better than the corresponding performance of the *greedy grSOM*. Again, the *greedy grSOM* engaged a smaller number of grid units in this problem than the *crisp SOM*.

Overall, our experimental work has produced significant statistical evidence that 1) the *GA optimized greedy grSOM* performs better than the *greedy grSOM*, and 2) the *greedy grSOM* performs better than the *crisp SOM*. Moreover, the corresponding standard deviation of either *greedy grSOM* variant was significantly smaller than the standard deviation of the *crisp SOM*.

The only substantial difference in the implementations of the *crisp SOM* and the *greedy grSOM* is that the former computes solely first order statistics (averages), whereas the latter computes higher order statistics in the training data. Therefore, it seems reasonable to conclude that the employment of higher order statistics in Kohonen's basic algorithm improves classification accuracy. Likewise, it seems reasonable to conclude that the employment of mass functions improves the classification accuracy further.

We have presented robust statistical evidence regarding the performance of three SOM variants in three benchmark classification problems in Tables 1, 3, and 5, where we report an average of ten experiments. Other algorithms typically report only a single best result in Tables 2 or 4. We point out that our best result in a series of ten experiments was never worse than the best result reported by another algorithm in a classification problem.

Additional advantages of a *grSOM* algorithm include, first, the induction of descriptive decision-making knowledge (fuzzy rules) from the training data and, second, the potential to cope rigorously with linguistic (fuzzy and/or interval) input data.

7.5 Technical remarks

Regarding the training time note that the *crisp SOM* in general took a few minutes to train. The *greedy grSOM* appeared to be eight to ten times slower than the *crisp SOM*. The need for longer training for the *greedy grSOM* is due to the employment of *FIN*s: first, it takes longer to compute a N-dimensional *FIN* vector than to compute a N-dimensional number vector average and, second, it takes longer to compute a distance d_K than to compute the L2

(Euclidean) distance. The genetic algorithm (GA) required an additional overhead of about 4 hours per (single) experiment. Despite its longer training, the better classification performance of a *greedy grSOM* variant may justify its employment in certain applications.
Regarding the speed of convergence it was recorded that after about 5,000 epochs more than 90% of the input data assigned to a *FIN* were data from a single class. The aforementioned percentage increased with time.

8. Conclusion

This work has proposed an extension of Kohonen's SOM, namely *greedy grSOM*, for inducing a distribution of Fuzzy Interval Numbers (*FIN*s) from real number samples. A *FIN* may represent a local probability distribution. The *greedy grSOM* was described as a fuzzy neural network for structure identification in linguistic (fuzzy) system modeling applications with emphasis in classification applications. The *GA optimized greedy grSOM* was also introduced for improving data classification based on genetically-tunable nonlinear mass functions. Extensive comparisons were shown with related work from the literature. Advantages of the proposed classifiers include: good performance demonstrated in three benchmark classification problems and induction of descriptive decision-making knowledge (fuzzy rules) from the training data. Despite its clearly better performance, a disadvantage of the *GA optimized greedy grSOM* is its long training time. The benchmark experiments in this work included solely trivial N-dimensional *FIN* inputs. Nevertheless, it was shown here that an employment of nontrivial N-dimensional *FIN* inputs is theoretically sound based on lattice theory mathematics.
This work has paved the way towards rigorous extensions of various clustering algorithms, e.g. Kohonen's SOM, *k*-means, etc., using linear convex combinations of *FIN*s. Knowledge acquisition and data mining applications could also be considered. Furthermore, future extensions to FIS modeling applications will consider multiple linguistic constraints.

Acknowledgements

We thank W. Pedrycz for bringing our attention to reference (Vuorimaa, 1994) regarding a fuzzy self-organizing map for FIS applications. This work has been supported in part by the third European framework programme: Operational Programme in Education and Initial Vocational Training II, under project Archimedes contract no. 04-3-001/1.

References

Atanassov, K.T. (1999). *Intuitionistic Fuzzy Sets: Theory and Applications*. Physica-Verlag.

Bertsekas, D., Nedic, A., & Ozdaglar, A.E. (2003). *Convex analysis and optimization*. Boston, MA: Athena Scientific.

Birkhoff, G. (1967). *Lattice Theory*. Providence, RI: American Mathematical Society, Colloquium Publications, **25**.

Bishop, C.M., Svensén, M., Williams, C.K.I. (1998). GTM: The generative topographic mapping. *Neural Computation*, **10**(1), 215-234.

Blake, C.L., & Merz, C.J. (1998). UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

Bortolan, G., & Pedrycz, W. (2002). Fuzzy descriptive models: An interactive framework of information granulation. *IEEE Transactions on Fuzzy Systems*, **10**(6), 743-755.

Carpenter, G.A., & Markuzon, N. (1998). ARTMAP-IC and medical diagnosis: Instance counting and inconsistent cases. *Neural Networks*, **11**(2), 323-336.

Chun, M.S., & Chang, H.-T. (2000). Application of neural networks incorporated with real-valued genetic algorithms in knowledge acquisition. *Fuzzy Sets and Systems*, **112**, 85-97.

Cottrell, M., Ibbou, S., & Letrémy, P. (2004). SOM-based algorithms for qualitative variables. *Neural Networks*, **17**(8-9), 1149-1167.

Dubois, D., & Prade, H. (1986). Fuzzy sets and statistical data. *European Journal of Operational Research*, **25**, 345-356.

Duda, R.O., Hart, P.E., & Stork, D.G. (2001). *Pattern classification*, 2nd edition. New York, NY: John Wiley & Sons, Inc.

Goodman, I.R., & Nguyen, H.T. (2002). Fuzziness and randomness. In C. Bertoluzza, M.A. Gil, & D.A. Ralescu (Eds.), *Statistical Modeling, Analysis and Management of Fuzzy Data*, **87** (pp. 3-21). Heidelberg, Germany: Physica-Verlag.

Hammer, B., & Villmann, T. (2002). Generalized relevance learning vector quantization. *Neural Networks*, **15**(8-9), 1059-1068.

Haritopoulos, M., Yin, H., & Allinson, N.M. (2002). Image denoising using self-organizing map-based nonlinear independent component analysis. *Neural Networks*, **15**(8-9), 1085-1098.

Hoffmann, F. (2001). Evolutionary algorithms for fuzzy control system design. *Proceedings of the IEEE*, **89**(9), 1318-1333.

Ishibuchi, H., Nakashima, T., & Murata, T. (1999). Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. *IEEE Transactions on Systems, Man and Cybernetics - Part B*, **29**(5), 601-618.

Ishigami, H., Fukuda, T., Shibata, T., & Arai, F. (1995). Structure optimization of fuzzy neural network by genetic algorithm. *Fuzzy Sets and Systems*, **71**, 257-264.

Itô, K. (1987). *Encyclopedic Dictionary of Mathematics*, second edition, the Mathematical Society of Japan, English translation. Cambridge, MA: The MIT Press.

Jang, J.-S. R., & Sun, C.-T. (1995). Neuro-fuzzy modeling and control. *Proceedings of the IEEE*, **83**(3), 378-406.

Joshi, A., Ramakrishman, N., Houstis, E.N., & Rice, J.R. (1997). On neurobiological, neuro-fuzzy, machine learning, and statistical pattern recognition techniques. *IEEE Transactions on Neural Networks*, **8**(1), 18-31.

Kaburlasos, V.G. (2002). Novel fuzzy system modeling for automatic control applications. *Proceedings 4th Intl. Conference on Technology & Automation* (268-275). Thessaloniki, Greece.

Kaburlasos, V.G. (2004). Fuzzy Interval Numbers (*FIN*s): Lattice theoretic tools for improving prediction of sugar production from populations of measurements. *IEEE Transactions on Systems, Man and Cybernetics – Part B*, 34(2), 1017-1030.

Kaburlasos, V.G., & Papadakis, S.E. (2004). *grSOM*: A granular extension of the self-organizing map for structure identification applications. *Proceedings of the IEEE International Conference on Fuzzy Systems* (789-794), Budapest, Hungary.

Kaburlasos, V.G., & Petridis, V. (2000). Fuzzy Lattice Neurocomputing (FLN) models. *Neural Networks*, **13**(10), 1145-1170.

Kaburlasos, V.G., & Petridis, V. (2002). Learning and decision-making in the framework of fuzzy lattices. In L.C. Jain, & J. Kacprzyk (Eds.), *New Learning Paradigms in Soft Computing*, **84** (pp. 55-96). Heidelberg, Germany: Physica-Verlag.

Kaburlasos, V.G., & Petridis, V. (2003). Improved prediction of industrial yield based on tools from a normed linear space of fuzzy interval numbers (FINs). *Proceedings of the 11th Mediterranean Conference on Control and Automation* (*MED'03*). Rhodes, Greece.

Kaburlasos, V.G., Petridis, V., Brett, P., & Baker, D. (1999). Estimation of the Stapes-Bone Thickness in Stapedotomy Surgical Procedure Using a Machine-Learning Technique. *IEEE Transactions on Information Technology in Biomedicine*, **3**(4), 268-277.

Karayiannis, N.B, & Bezdek, J.C. (1997). An integrated approach to fuzzy learning vector quantization and fuzzy *c*-means clustering. *IEEE Transactions on Fuzzy Systems*, **5**(4), 622-628.

Kazarlis, S.A., Papadakis, E., Theocharis, J.B., & Petridis, V. (2001). A Micro-Genetic Algorithms as Generalized Hill-Climbing Operators for GA Optimazation. *IEEE Transactions on Evolutionary Computation*, **5**(3), 204-217.

Kecman, V. (2001). *Learning and Soft Computing*. Cambridge, Massachusetts: The MIT Press.

Kohonen, T. (1995). *Self-organizing maps*. Berlin, Germany: Springer.

Kohonen, T. (1996). Self-organizing maps of symbol strings. *Technical report A42*, Laboratory of Computer and Information Science, Helsinki University of Technology, Finland.

Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paatero, V., & Saarela, A. (2000). Self organization of a massive document collection. *IEEE Trans. on Neural Networks*, **11**(3), 574-585.

Kohonen, T., & Somervuo, P. (1998). Self-organizing maps of symbols strings. *Neurocomputing*, **21**(1-3), 19-30.

Körner, R., & Näther, W. (2002). On the variance of random fuzzy variables. In C. Bertoluzza, M.A. Gil, & D.A. Ralescu (Eds.), *Statistical Modeling, Analysis and Management of Fuzzy Data*, **87** (pp. 25-42). Heidelberg, Germany: Physica-Verlag.

Krishnapuram, R., Keller, J.M. (1993). A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems*, **1**(2), 98-110.

Kuncheva, L.I. (1996). On the equivalence between fuzzy and statistical classifiers. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, **4**(3), 245-253.

Leng, G., McGinnity, T.M., Prasad, G. (2005). An approach for on-line extraction of fuzzy rules using a self-organising fuzzy neural network. *Fuzzy Sets and Systems*, **150**(2), 211-243.

Lin, C.J., & Lin, C.T. (1997). An ART-based fuzzy adaptive learning control network. *IEEE Transactions on Fuzzy Systems*, **5**(4), 477-496.

Mamdani, E.H., & Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *Intl. J. of Man-Machine Studies*, **7**, 1-13.

Mitra, S., & Hayashi, Y. (2000). Neuro-fuzzy rule generation: survey in soft computing framework. *IEEE Transactions on Neural Networks*, **11**(3), 748-768.

Mitra, S., & Pal, S.K. (1994). Self-organizing neural network as a fuzzy classifier. *IEEE Transactions on Systems, Man and Cybernetics*, **24**(3), 385-399.

Mitra, S., & Pal, S.K. (1996). Fuzzy self-organization, inferencing, and rule generation. *IEEE Transactions on Systems, Man and Cybernetics – Part A*, **26**(5), 608-620.

Pal, N.R., Bezdek, J.C., & Tsao, E.C.-K. (1993). Generalized clustering networks and Kohonen's self-organizing scheme. *IEEE Transactions on Neural Networks*, **4**(4), 549-557.

Papadakis, S., & Theocharis, J. (1996). A novel genetic training algorithm with application to fuzzy neural networks. *Proceedings of the IASTED International Conference on System Modeling Identification and Control* (125-128), Insburg, Austria.

Papadakis, S.E., & Theocharis, J.B. (2002). A GA-based fuzzy modeling approach for generating TSK models. *Fuzzy Sets and Systems,* **131**(2), 121-152.

Papadimitriou, S., Mavroudi, S., Vladutu, & Bezerianos, A. (2001). Ischemia detection with a self-organizing map supplemented by supervised learning. *IEEE Transactions on Neural Networks*, **12**(3), 503-515.

Pedrycz, W. (2002). Granular networks and granular computing. In L.C. Jain, & J. Kacprzyk (Eds.), *New Learning Paradigms in Soft Computing*, **84** (pp. 30-54). Heidelberg, Germany: Physica-Verlag.

Pedrycz, W., & Bargiela, A. (2002). Granular clustering: A granular signature of data. *IEEE Transactions on Systems, Man and Cybernetics – Part B*, **32**(2), 212-224.

Peltonen, J., Klami, A., & Kaski, S. (2004). Improved learning of Riemannian metrics for exploratory analysis. *Neural Networks*, **17**(8-9), 1087-1100.

Petridis, V., & Kaburlasos, V.G. (1998). Fuzzy Lattice Neural Network (FLNN): A hybrid model for learning. *IEEE Transactions on Neural Networks*, **9**(5), 877-890.

Petridis, V., & Kaburlasos, V.G. (2001). Clustering and classification in structured data domains using Fuzzy Lattice Neurocomputing (FLN). *IEEE Transactions on Knowledge and Data Engineering*, **13**(2), 245-260.

Petridis, V., & Kaburlasos, V.G. (2003). FINkNN: A fuzzy interval number k-nearest neighbor classifier for prediction of sugar production from populations of samples. *Journal of Machine Learning Research*, **4**(Apr), 17-37.

Polani, D. (1999). On the optimization of self-organizing maps by genetic algorithms. In E. Oja, & S. Kaski (Eds.), *Kohonen Maps*, (pp. 157-169). Amsterdam, NL: Elsevier.

Pomares, H., Rojas, I., González, J., & Prieto, A. (2002). Structure identification in complex rule-based fuzzy systems. *IEEE Transactions on Fuzzy Systems*, **10**(3), 349-359.

Principe, J.C., Wang, L., & Motter, M.A. (1998). Local dynamic modeling with self organizing maps and applications to nonlinear system identification and control. *Proceedings of the IEEE*, **86**(11), 2240 –2258.

Riesz, F. (1928). Sur la décomposition des opérations fonctionnelles linéaires. *Atti del Congresso Internazionale dei Matematici*, **3**, 143-148.

Ritter, H. (1999). Self-organizing maps on non-euclidean spaces. In E. Oja, & S. Kaski (Eds.), *Kohonen Maps*, (pp. 97-109). Amsterdam, NL: Elsevier.

Seo, A., & Obermayer, K. (2004). Self-organizing maps and clustering methods for matrix data. *Neural Networks*, **17**(8-9), 1211-1229.

Setnes, M. (2000). Supervised fuzzy clustering for rule extraction. *IEEE Transactions on Fuzzy Systems*, **8**(4), 416-424.

Shimojima, K., Fukuda, T., & Hasegawa, Y. (1995). Self-tuning fuzzy modeling with adaptive membership function, rules, and hierarchical structure based on genetic algorithms. *Fuzzy Sets and Systems*, **71**, 295-309.

Somervuo, P.J. (2004). Online algorithm for the self-organizing map of symbol strings. *Neural Networks*, **17**(8-9), 1231-1239.

Sugeno, M., & Kang, G.T. (1988). Structure identification of fuzzy model. *Fuzzy Sets and Systems*, **28**(1), 15-33.

Tagaki, T., and Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, **15**(1), 116-132.

Tsao, E.C.-K., Bezdek, J.C., & Pal, N.R. (1994). Fuzzy Kohonen clustering networks. *Pattern recognition*, **27**(5), 757-764.

Verbeek, J.J., (2005). *Mixture Models for Clustering and Dimension Reduction*. PhD dissertation, University of Amsterdam, The Netherlands. < http://staff.science.uva.nl/~jverbeek/thesis.html>

Verbeek, J.J., Vlassis, N., & Kröse, B.J.A. (2005). Self-organizing mixture models. *Neurocomputing*, **63**, 99-123.

Vuorimaa, P. (1994). Fuzzy self-organizing map. *Fuzzy Sets and Systems*, **66**(2), 223-231.

Vulikh, B.C. (1967). *Introduction to the theory of partially ordered vector spaces*. Gronigen.

Yen, J., & Wang, L. (1998). Application of statistical information criteria for optimal fuzzy model construction. *IEEE Transactions on Fuzzy Systems*, **6**(3), 362-372.

Zadeh, L.A. (1965). Fuzzy sets. *Information and Control*, **8**, 338-353.

Zadeh, L.A. (1978). Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, **1**(1), 3-28.

Zadeh, L.A. (1999). From computing with numbers to computing with words. From manipulation of measurements to manipulation of perceptions. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, **46**(1), 105 –119.

Table 1
FISHER IRIS benchmark: Performance of three SOM variants in 10 different random data partitions.

		% classification accuracy on the Training data		Testing data		no. grid units engaged	
Data Partition	Algorithm	average	stdv	average	stdv	average	stdv
90% for training / 10% for testing	*crisp SOM*	92.88	4.12	93.33	5.44	16.0	0.00
	greedy grSOM with $m(x)=1$	95.55	1.78	96.66	3.51	10.1	1.59
	greedy grSOM GA optimized	97.77	0.92	98.70	2.81	10.1	1.59
66% for training / 34% for testing	*crisp SOM*	93.30	3.65	91.00	6.20	16.0	0.00
	greedy grSOM with $m(x)=1$	95.80	1.54	95.60	2.27	9.0	1.63
	greedy grSOM GA optimized	97.70	1.25	97.60	1.26	9.0	1.63

Table 2
Performance of various algorithms on the WINE RECOGNITION benchmark data set for classification. The algorithms have been arranged in a decreasing order of success.

Algorithm	% classification accuracy on the testing data
Resilient Backpropagation (Rprop)	100.00
Regularized Discriminant Analysis (RDA)	100.00
Quadratic Discriminant Analysis (QDA)	99.40
Linear Discriminant Analysis(LDA)	98.90
Learning Vector Quantization(LVQ1)	96.20
1-Nearest-Neighbor (1NN)	96.10
C4.5	96.00
ID3	95.20
Bprop	95.18
AutoClass	84.84
CLUSTER	84.09
Simpson's algorithm	82.57

Table 3
WINE RECOGNITION benchmark: Performance of three SOM variants in 10 different random data partitions.

		% classification accuracy on the				no. grid units engaged	
		Training data		Testing data			
Data Partition	Algorithm	average	stdv	average	stdv	average	stdv
90% for training / 10% for testing	*crisp SOM*	95.06	1.70	91.10	6.52	16.0	0.00
	greedy grSOM with $m(x)=1$	98.49	0.52	98.33	2.68	10.0	1.15
	greedy grSOM GA optimized	98.99	0.79	99.40	1.76	10.0	1.15
66% for training / 34% for testing	*crisp SOM*	96.35	1.55	95.83	3.26	16.0	0.00
	greedy grSOM with $m(x)=1$	98.81	0.43	97.33	1.61	9.1	2.13
	greedy grSOM GA optimized	99.15	1.06	99.30	0.86	9.1	2.13

Table 4
Performance of various algorithms on the CLEVELAND HEART DISEASE benchmark data set for classification. The algorithms have been arranged in a decreasing order of success.

Algorithm	% classification accuracy on the testing data
FLNtf (10-fold cross-validation)	79.34
Logistic regression	79.00
Conceptual clustering (CLASSIT)	78.90
ARTMAP-IC	78.00
FLNtf (keep-250-in)	77.88 (*)
Discriminant analysis	77.00
Instance based prediction (NTgrowth)	77.00
Instance based prediction (C4)	74.80
Fuzzy ARTMAP	74.00
kNN	67.00

(*) Average for 100 random data partitions

Table 5
CLEVELAND HEART DISEASE benchmark: Performance of three SOM variants in 10 different random data partitions.

		% classification accuracy on the				no. grid units	
		Training data		Testing data		engaged	
Data Partition	Algorithm	average	stdv	average	stdv	average	stdv
90% for training / 10% for testing	*crisp SOM*	78.74	1.78	78.76	10.94	16.0	0.00
	greedy grSOM with $m(x)=1$	81.49	0.93	81.99	4.21	14.0	1.41
	greedy grSOM GA optimized	84.29	1.87	84.70	1.72	14.0	1.41
250 data for training / 53 data for testing	*crisp SOM*	79.70	2.85	75.20	5.73	16.0	0.00
	greedy grSOM with $m(x)=1$	82.55	2.61	79.90	3.63	14.5	0.97
	greedy grSOM GA optimized	84.65	1.56	83.20	1.62	14.5	0.97

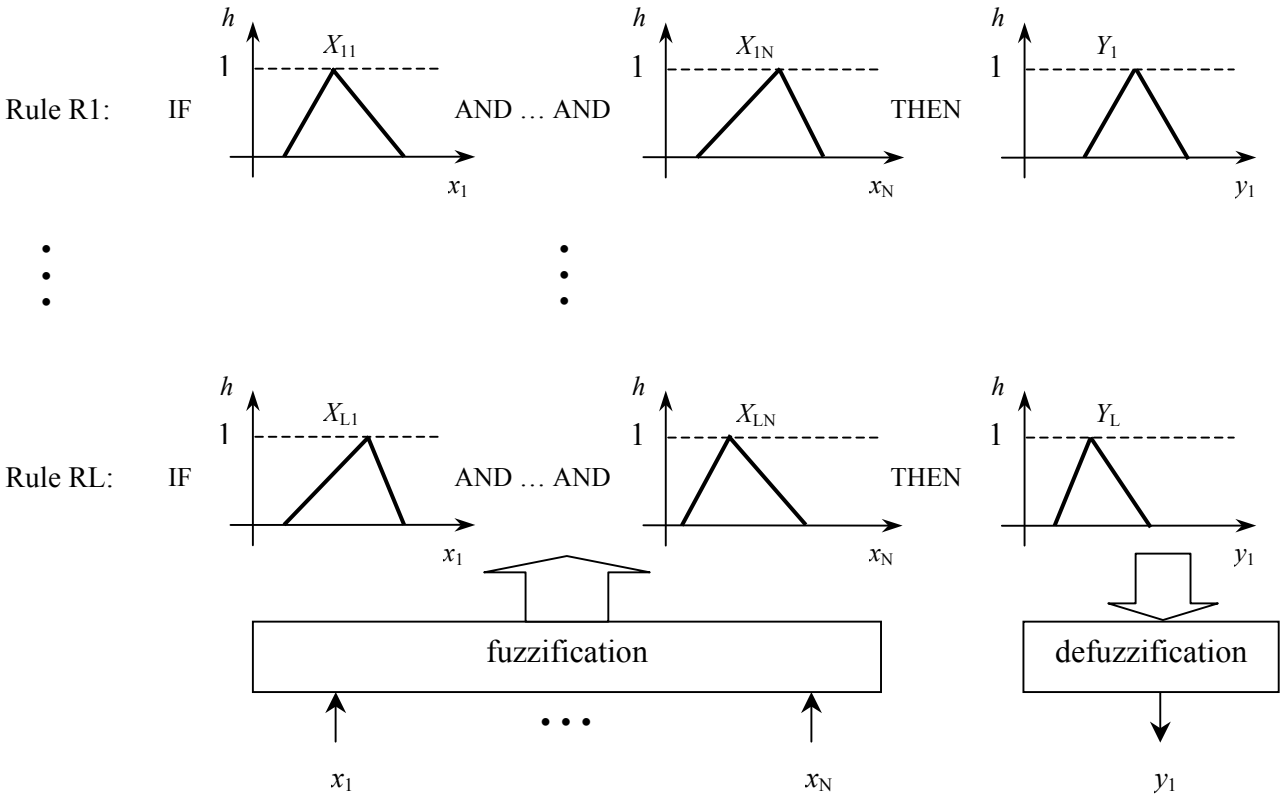
Rule R1:
IF
AND … AND
THEN
h
1
X_{11}
x_1
X_{1N}
x_N
Y_1
y_1
Rule RL:
IF
AND … AND
THEN
X_{L1}
X_{LN}
Y_L
fuzzification
defuzzification
x_1
x_N
y_1

Fig. 1

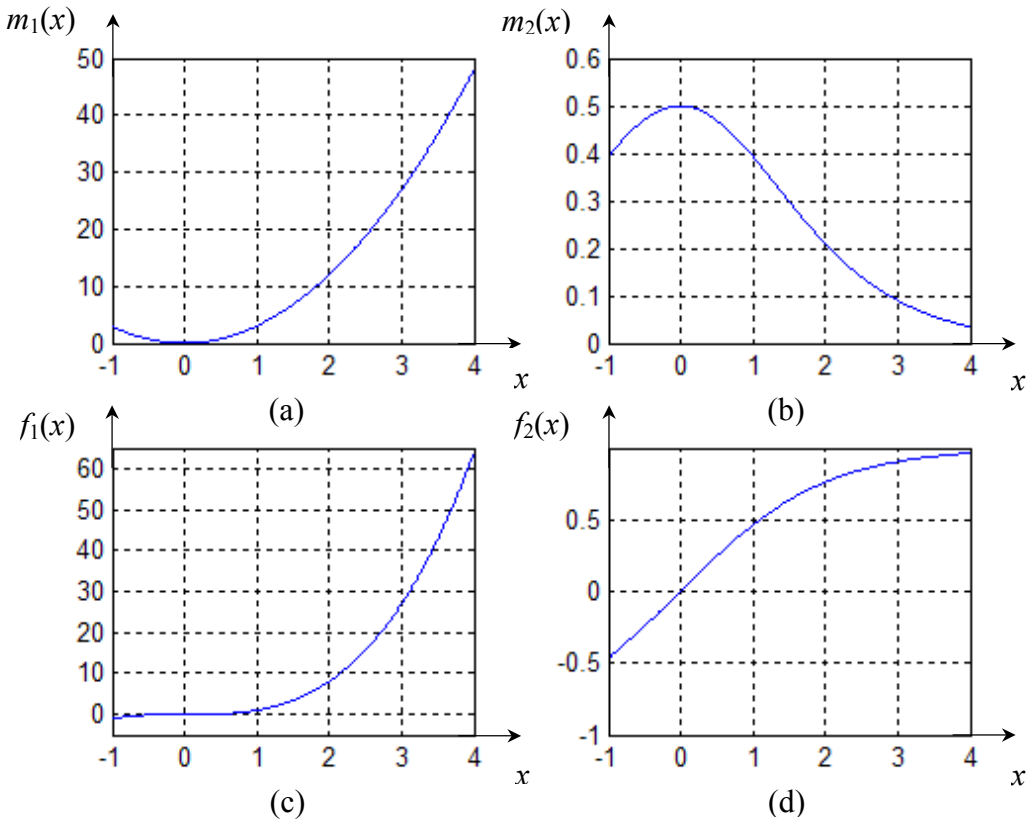
$m_1(x)$
50
40
30
20
10
0
-1 0 1 2 3 4
x
(a)
$m_2(x)$
0.6
0.5
0.4
0.3
0.2
0.1
0
-1 0 1 2 3 4
x
(b)
$f_1(x)$
60
50
40
30
20
10
0
-1 0 1 2 3 4
x
(c)
$f_2(x)$
0.5
0
-0.5
-1
-1 0 1 2 3 4
x
(d)

Fig. 2

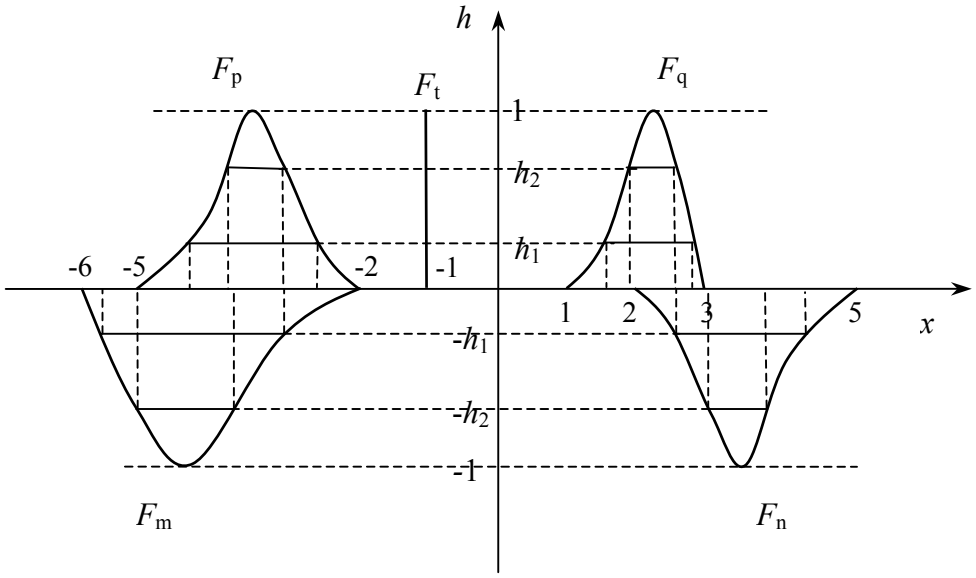

Fig. 3

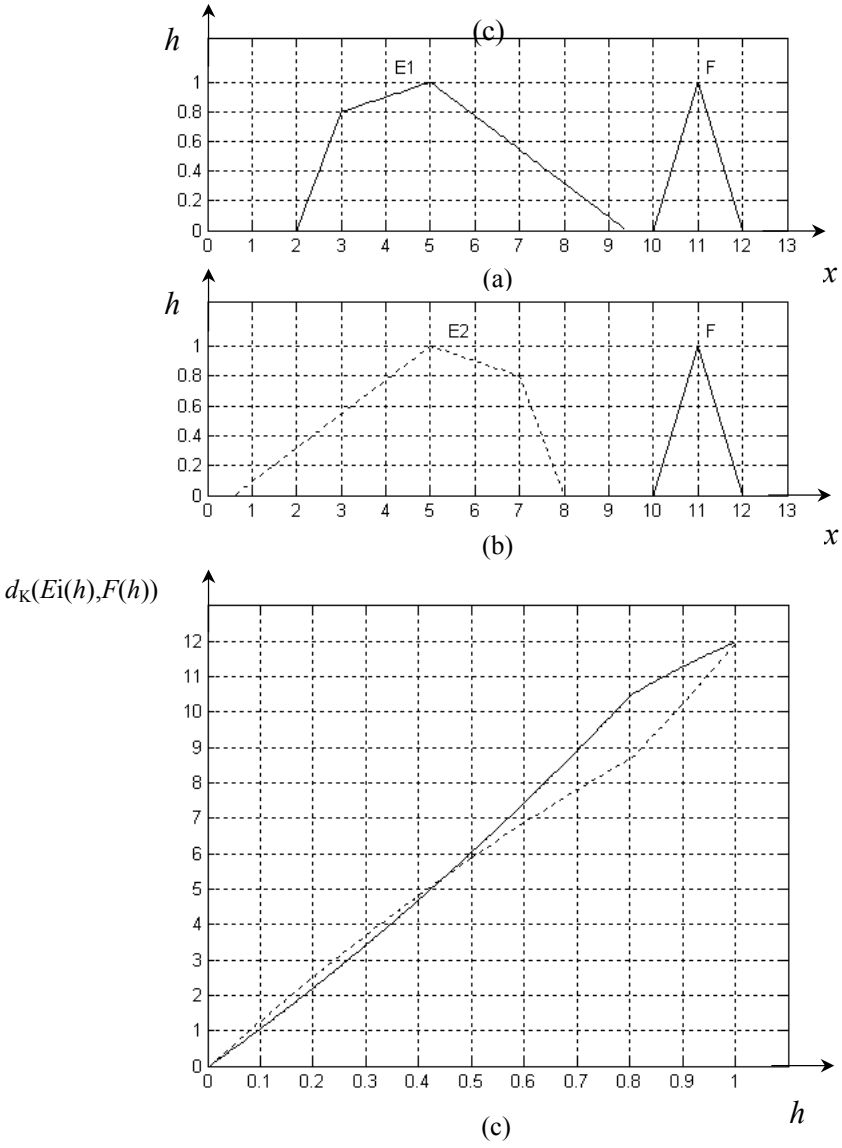

Fig. 4

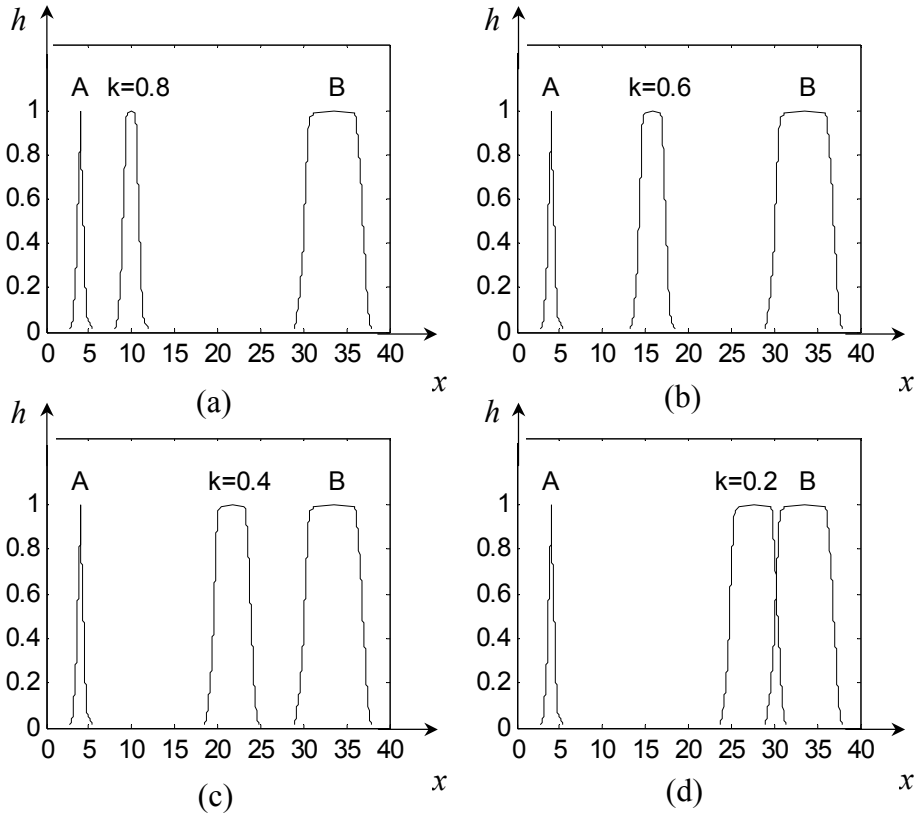

Fig. 5

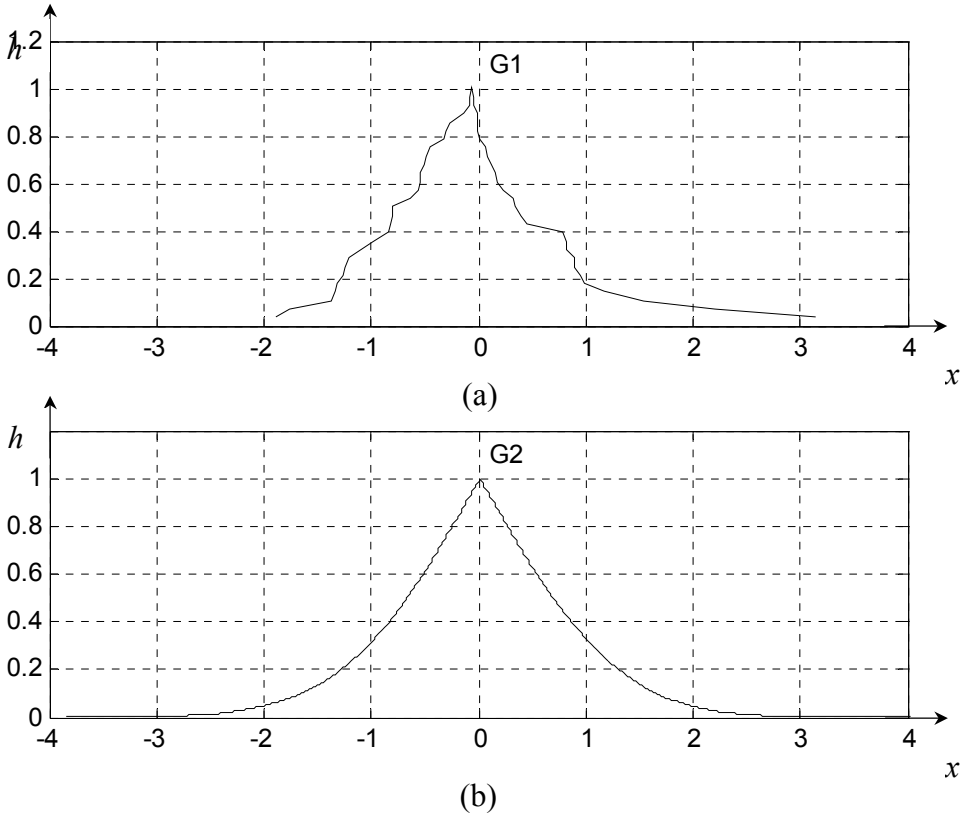
h
G1
1.2
1
0.8
0.6
0.4
0.2
0
-4
-3
-2
-1
0
1
2
3
4
x
(a)
h
G2
1
0.8
0.6
0.4
0.2
0
-4
-3
-2
-1
0
1
2
3
4
x
(b)

Fig. 6

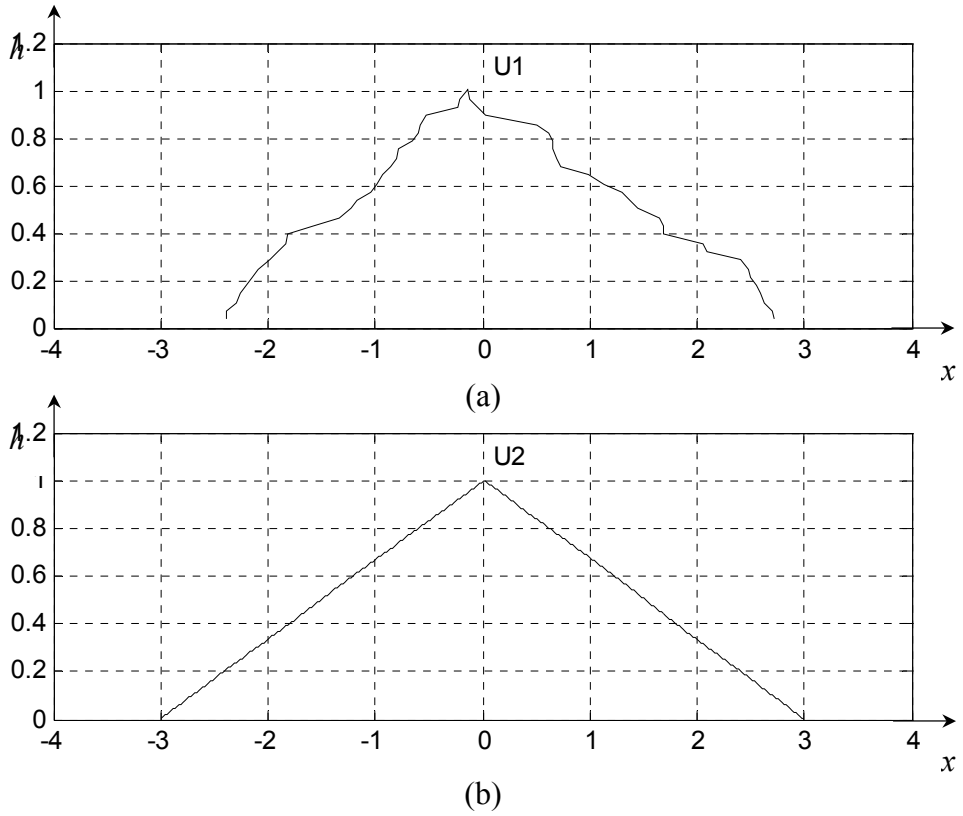
μ
1.2
1
0.8
0.6
0.4
0.2
0
U1
-4
-3
-2
-1
0
1
2
3
4
x
(a)
μ
1.2
1
0.8
0.6
0.4
0.2
0
U2
-4
-3
-2
-1
0
1
2
3
4
x
(b)

Fig. 7

Step-1: Define the dimensions I and J of a two-dimensional grid of I×J units/neurons. Each unit can store both a N-tuple weight $W_{ij} \in \mathsf{F}^N$, and a label L_{ij}, i=1,…,I, j=1,…,J; the latter indicates the category of the unit.

Step-2: Initialize randomly the weight W_{ij} of each unit from the training data set.

Repeat steps 3 and 4 below for a user-defined integer number *Nepochs* of epochs.

Step-3: For each training input datum $x_k \in \mathsf{F}^N$, k=1,…,n do

a) Calculate the Minkowski metric distance $d_1(x_k, W_{ij})$, i=1,…,I, j=1,…,J.
b) Competition among the I×J units in the grid: Winner is the unit 'pq' whose weight W_{pq} is the nearest to x_k.
c) Assign the training input x_k equally to both the winner unit 'pq' and to all the units in the neighborhood $N_{pq}(t)$ of the winner.

Step-4: For each unit 'ij', i=1,…,I, j=1,…,J in the grid use algorithm CALFIN to compute the new weight value W'_{ij} based on the data assigned to unit 'ij' in Step-3 of the current epoch.

Step-5: To each unit 'ij', i=1,…,I, j=1,…,J in the grid attach the label of the category, which provided the majority of the input training data to unit 'ij' during all epochs.

Fig. 8

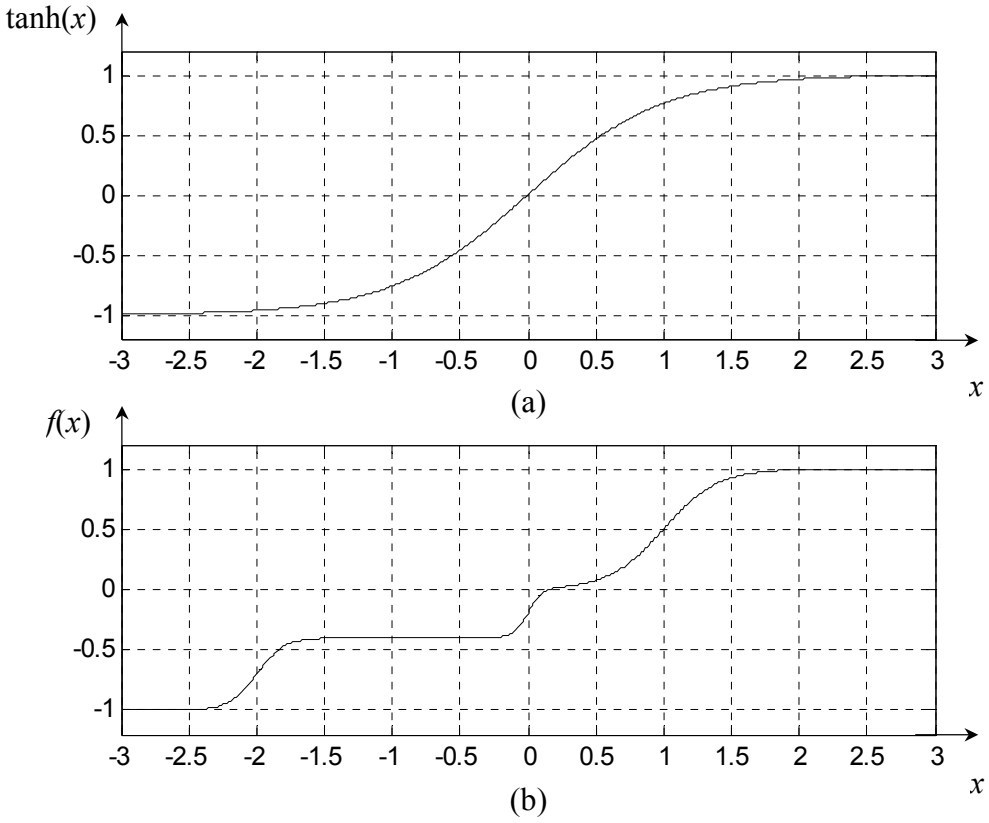
tanh(x)
1
0.5
0
-0.5
-1
-3
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
3
x
(a)
f(x)
1
0.5
0
-0.5
-1
-3
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
3
x
(b)

Fig. 9

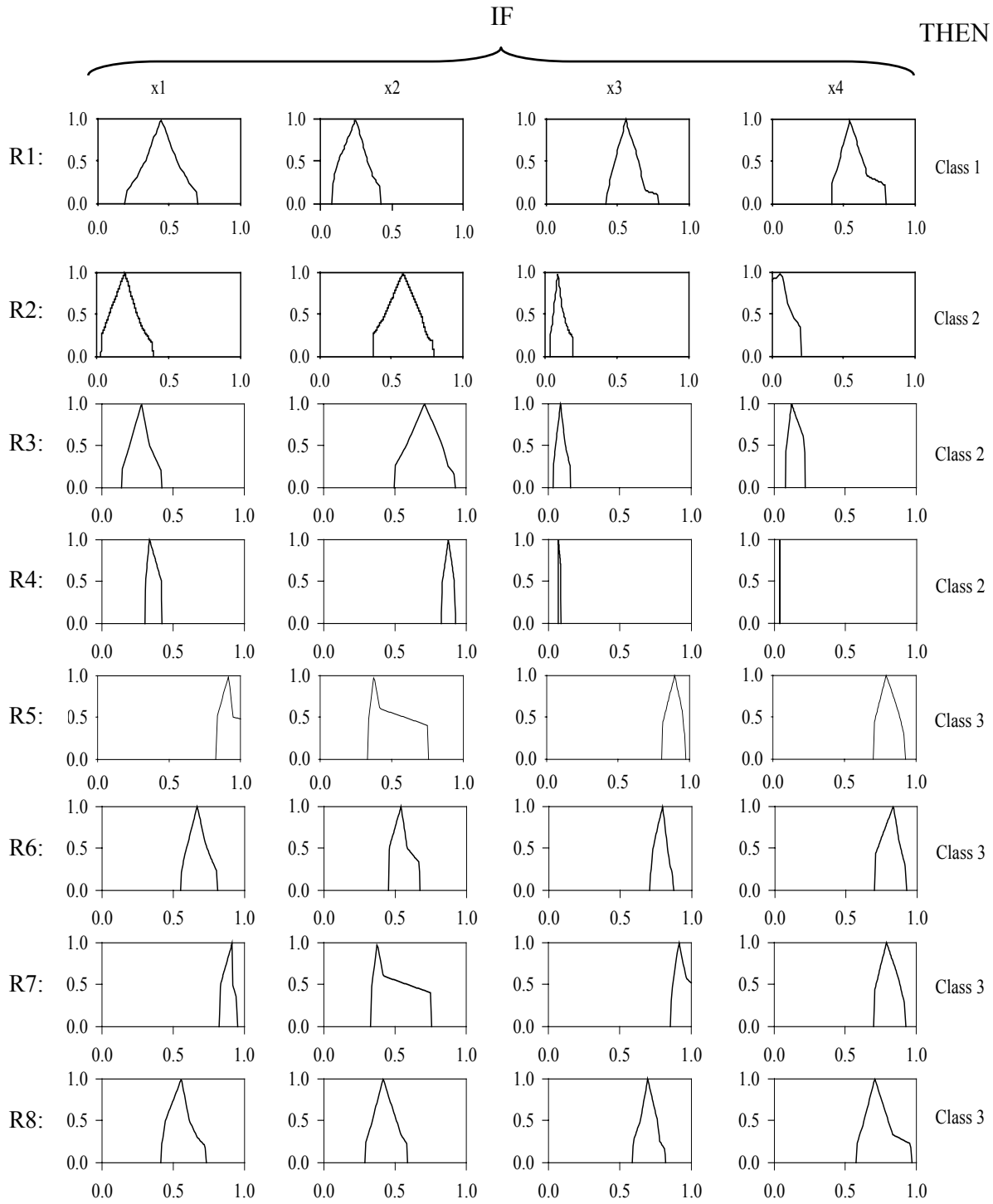
IF
THEN
x1
x2
x3
x4
R1:
R2:
R3:
R4:
R5:
R6:
R7:
R8:
Class 1
Class 2
Class 2
Class 2
Class 3
Class 3
Class 3
Class 3
1.0
0.5
0.0
0.0
0.5
1.0

Fig. 10

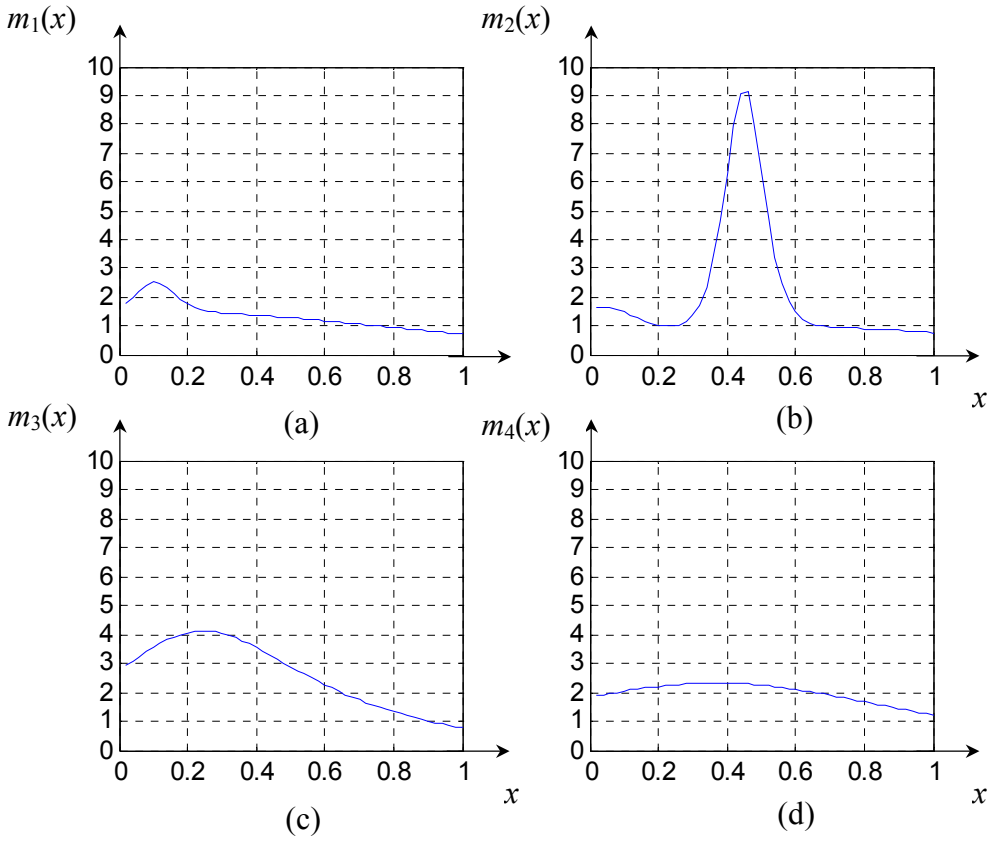

Fig. 11

FIGURE LEGENDS

Fig. 1 A Mamdani type Fuzzy Inference System (FIS) with N inputs $x_1,\ldots,x_N$, one output y_1, and L fuzzy rules R1,…,RL. The above FIS, including both a fuzzification and a defuzzification procedure, is used for implementing a function f: $\mathsf{R}^N \rightarrow \mathsf{R}$.

Fig. 2 (a) Symmetric mass function $m_1(x)= 3x^2$.

(b) Symmetric mass function $m_2(x)= \dfrac{2e^{-x}}{(1+e^{-x})^2}$.

(c) The *cubic* strictly increasing function $f_1(x)= x^3$ is an integral of mass function $m_1(x)$.

(d) The *logistic* strictly increasing function $f_2(x)= \dfrac{2}{1+\mathrm{e}^{-x}} - 1$ is an integral of mass function $m_2(x)$.

Fig. 3 Positive *FIN*s F_p, F_q include positive generalized intervals $F_p(h)$, $F_q(h)$, $h \in (0,1]$; negative *FIN*s F_m, F_n include negative generalized intervals $F_m(h)$, $F_n(h)$, $h \in (0,1]$; the trivial *FIN* F_t includes trivial generalized intervals $[-1,-1]^h$, $h \in (0,1]$.

Fig. 4 (a) *FIN*s E1 and F.

(b) *FIN*s E2 and F.

(c) The metric distance functions of generalized interval $F(h)$ from either generalized interval E1(h) (solid line) or generalized interval E2(h) (dotted line) versus h for $h \in (0,1]$. The area underneath a curve equals the corresponding metric distance between two *FIN*s. It turns out to be $d_K(\mathrm{E1},\mathrm{F}) \approx 6.18 > 5.81 \approx d_K(\mathrm{E2},\mathrm{F})$.

Fig. 5 The convex combination $kA+(1-k)B$ for various values of k including (a) $k= 0.8$, (b) $k= 0.6$, (c) $k= 0.4$, (d) $k= 0.2$. Notice that the sum *FIN* '$kA+(1-k)B$' is progressively a combination of both the location and the shape of *FIN*s A and B.

Fig. 6 A *FIN* in this figure was computed by algorithm CALFIN from random numbers generated according to the normal (Gaussian) probability density function N(0,1) with *mean* 0 and *standard deviation* 1. More specifically,

(a) 50 random numbers were used to compute *FIN* G1.

(b) 10,000 random numbers were used to compute *FIN* G2.

Fig. 7 A *FIN* in this figure was computed by algorithm CALFIN from random numbers drawn according to the uniform probability density function over the range [-3,3]. More specifically,

(a) 50 random numbers were used to compute *FIN* U1.

(b) 10,000 random numbers were used to compute *FIN* U2.

Fig. 8 The training phase of the *greedy grSOM* algorithm for supervised clustering. Clusters are located in the data by N-tuples of *FIN*s, i.e. grid unit weights, so as to cover the training data domain. Finally, a category label is attached to a grid unit.

Fig. 9 (a) The saturated function tanh(x).

(b) Function $f(x) = 0.3\tanh((x+2)/0.2) + 0.2\tanh(x/0.1) + 0.5\tanh((x-1)/0.4)$. The parameter values of a component function 'tanh(.)' determine the location, scale, and the height of the corresponding component function.

Fig. 10 Eight rules R1…R8 induced from the IRIS benchmark data in a 90%-10% data partition. A rule's antecedent (IF part) is the conjunction of four fuzzy statements and a rule's consequent (THEN part) is a class label. One rule (R1) was induced for class 1, which is linearly separable from the other two classes; three and four rules were induced, respectively, for the nonlinearly separable classes 2 and 3.

Fig. 11 Mass functions (a) $m_1(x)$, (b) $m_2(x)$, (c) $m_3(x)$, and (d) $m_4(x)$ computed by a Genetic Algorithm (GA) on the four data dimensions x_1, x_2, x_3, and x_4, respectively, for the IRIS benchmark experiment whose induced rules are shown in Fig.10.