Abstracts of the FLLL/SCCH Master and PhD Seminar

Room 010, Software Park Hagenberg

November 19, 2004

Software Competence Center Hagenberg
Hauptstrasse 99
A-4232 Hagenberg
Tel. +43 7236 3343 800
Fax +43 7236 3343 888
www.scch.at

Fuzzy Logic Laboratorium Linz-Hagenberg
Hauptstrasse 99
A-4232 Hagenberg
Tel. +43 7236 3343 431
Fax +43 7236 3343 434
www.flll.jku.at

Program

Session 1 (Chair: Edwin Lughofer) 8:30–10:30

8:30 Koen Maes, Bernard De Baets:
T-norm solutions of a system of functional equations

9:00 Andrea Mesiarová:
k-Lipschitz t-norms

9:30 Dragan Jočić:
Conditionally distributivity of a uni-norm with respect to a triangular conorm

10:00 Susanne Saminger, Koen Maes:
Aggregation of reciprocal relations

10:30 Coffee Break

Session 2 (Chair: Ulrich Bodenhofer) 10:50–12:50

10:50 Thomas Biringer:
Knowledge based methods in management

11:20 Thomas Natschläger, Nils Bertschinger, Robert Legenstein:
Real-time computations at the edge of chaos and self-organized criticality in recurrent neural networks

11:50 Ester Van Broekhoven, Veronique Adriaenssens, Bernard De Baets:
Evaluation and interpretability-preserving optimization of a fuzzy ordered classifier

12:20 Edwin Lughofer, Erich Peter Klement:
FLEXFIS: A variant for incremental learning of Takagi-Sugeno fuzzy systems

12:50 Lunch Break

Session 3 (Chair: Edwin Lughofer) 14:00–16:00

14:00 Ivana Štajner-Papuga:
Some generalizations of the pseudo-Laplace transform

14:30 Leila Muresan, Bettina Heise, Jan Kybic, Erich Peter Klement:
Quantitative analysis of microarray images

15:00 Aleksandar Takači:
pFCSP and implementation of priority queries in FSQL

15:30 Roland Richter:
The data/view model: application examples

T-norm solutions of a system of functional equations

K. Maes and B. De Baets

Department of Applied Mathematics, Biometrics and Process Control
Ghent University, Coupure links 653, B-9000 Gent, Belgium
{Koen.Maes,Bernard.DeBaets}@UGent.be

Abstract

In fuzzy set theory, the fuzzification of a crisp concept is not seldom rooted in a straightforward adjustment of the disjunctive or conjunctive Boolean normal form of the underlying mathematical expression. However, the fuzzified normal forms obtained can rarely be considered as true normal forms in an extended logic or algebra. They are to be considered as functions, defined on $[0,1]^n$, for some $n \in \mathbb{N}_0$, and taking values in the support $[0,1]$ of a BL-algebra $([0,1], \vee, \wedge, T, I_T, 0, 1)$, with T a continuous t-norm and I_T its residual implicator. In this paper, we figure out to what extent the mutual distance between both fuzzified normal forms depends on the original Boolean function. Special attention is drawn to the Łukasiewicz triplet, as it is the only continuous De Morgan triplet for which the difference between both fuzzified normal forms is independent of the underlying Boolean function.

T-norm, t-conorm, De Morgan triplet, Łukasiewicz triplet, disjunctive normal form, conjunctive normal form, Cauchy equation

1 Introduction

Attempting to capture imprecision associated with the combination of concepts, Türkşen introduced interval-valued fuzzy sets based on a straightforward fuzzification (interpreting $\wedge$ as a t-norm T, $\vee$ as a t-conorm S and $'$ as a negator N) of the Boolean disjunctive and conjunctive normal forms of the original Boolean expression [14, 16]. He claims that by doing so, interpretation-dependent difficulties can be avoided. However, Türkşen's fuzzified normal forms cannot be seen as true normal forms. At best, if we work with the Kleene algebra $([0,1], \min, \max, \mathcal{N}, 0, 1)$, they yield lower and upper approximations of a given wff[1] over that Kleene algebra. In general, these disjunctive and conjunctive fuzzified normal forms are $[0,1]$-valued functions on $[0,1]^n$, for some $n \in \mathbb{N}_0$. They sometimes

[1] well formed formula

provide a kind of standard fuzzification procedure. The reason for this lies in the observation that the crisp concepts themselves are often mathematically expressed by means of their disjunctive or conjunctive normal form. For example, when constructing fuzzy preference structures (P, I, J), researchers made intensive use of the fuzzified disjunctive normal forms of the original crisp binary relations P, I and J [4]. Until now, little is known about the relationships between the fuzzified normal forms. To get a better understanding of the true meaning of the fuzzified normal forms, we should first explore more profoundly their comparability. In particular we deal with a system of functional equations that originated by imposing some functional independence on the difference between both fuzzified normal forms. This system of functional equations has a unique solution when working with continuous t-norms and t-conorms.

2 A system of functional equations

For the sake of brevity, we denote from now on $\{0,1\}$ (resp. $[0,1]$) by B (resp. I). In the Boolean algebra $\mathcal{B} = (B, \vee, \wedge, ', 0, 1)$, every function $f : B^n \to B$ can be represented by its unique disjunctive ($D_{\mathcal{B}}$) and conjunctive ($C_{\mathcal{B}}$) normal form. Replacing $(\wedge, \vee, ')$ by a triplet (T, S, N), with N an involutive negator, results in a straightforward fuzzification of these Boolean normal forms. The corresponding disjunctive and conjunctive fuzzified normal forms are denoted by D_F and C_F. For each n-ary Boolean function f, we obtain two $I^n \to I$ mappings $D_F(f)$ and $C_F(f)$:

$$\begin{aligned} D_F(f)(\mathbf{x}) &= S\{f(\mathbf{e})\, T(\mathbf{x}^{\mathbf{e}}) \mid \mathbf{e} \in B^n\}, \\ C_F(f)(\mathbf{x}) &= T\left\{\left[(1 - f(\mathbf{e}))\, S\left(\mathbf{x}^{(\mathbf{e}^{\mathbf{0}})}\right)^N\right]^N \mid \mathbf{e} \in B^n\right\}, \end{aligned}$$

where $\mathbf{x} \in I^n$, $\mathbf{0} = (0, ..., 0)$, $\mathbf{x}^{\mathbf{e}} = (x_1^{e_1}, ..., x_n^{e_n})$, $x_i^{e_i} = x_i$ if $e_i = 1$ and $x_i^{e_i} = x_i^N$ if $e_i = 0$. In case (T, S, N) is a De Morgan triplet, the conjunctive fuzzified normal form can also be written as

$$\begin{aligned} C_F(f)(\mathbf{x}) &= T\{[(1 - f(\mathbf{e}))\, T(\mathbf{x}^{\mathbf{e}})]^N \mid \mathbf{e} \in B^n\} \\ &= [S\{(1 - f(\mathbf{e}))\, T(\mathbf{x}^{\mathbf{e}}) \mid \mathbf{e} \in B^n\}]^N. \end{aligned}$$

Note that the use of an involutive negator is indispensable to define these fuzzified normal forms unequivocally [9].

By definition, the difference between both fuzzified normal forms is given by

$$\begin{aligned} &C_F(f)(\mathbf{x}) - D_F(f)(\mathbf{x}) \\ &= T\left\{\left[(1 - f(\mathbf{e}))\, S\left(\mathbf{x}^{(\mathbf{e}^{\mathbf{0}})}\right)^N\right]^N \mid \mathbf{e} \in B^n\right\} - S\{f(\mathbf{e})\, T(\mathbf{x}^{\mathbf{e}}) \mid \mathbf{e} \in B^n\}, \end{aligned} \tag{1}$$

for any Boolean function f and any $\mathbf{x} \in I^n$. We are now looking for those triplets (T, S, N) for which $C_F(f)(\mathbf{x}) - D_F(f)(\mathbf{x})$ is only a function of the variable $\mathbf{x} \in I^n$ (i.e. independent

of the Boolean function f). In particular these triplets (T, S, N) solve the system of functional equations, obtained by putting, for every $\mathbf{x} \in I^n$, the 2^{2^n} different expressions (1) on a par. Let a_N be the fixpoint of the involutive negator N. To solve the system of functional equations it is sufficient to look for those triplets that equalize the following expressions

$$S(x, y^N) - T(x^N, y^N)\,, \tag{2}$$

$$S(x, y) - T(x^N, y)\,, \tag{3}$$

$$T(S(x, y^N), S(x, y))\,, \tag{4}$$

$$1 - S(T(x^N, y), T(x^N, y^N))\,, \tag{5}$$

for any $(x, y) \in [0, a_N]^2$, $x \leq y$ [9]. Afterwards we can verify wheter the solutions of this 2-dimensional problem also fulfill the original system (1). From now on, if we deal with the system of equalities $(2) = (3) = (4) = (5)$, we take for granted that $(x, y) \in [0, a_N]^2$ and $x \leq y$.

3 Continuous solutions

Taking a closer look at expressions (2)–(5) we see that $(2) = (3)$ and $(4) = (5)$ hold when $N = \mathcal{N}$ and $(T, S, N) = \langle T, S, N\rangle$ (i.e. (T, S, N) is a De Morgan triplet). However, for both equalities, the properties $N = \mathcal{N}$ and $(T, S, N) = \langle T, S, N\rangle$ get entangled. Although various attempts to unlink them failed, we were in some sense able to emphasize their mutual connection: one property cannot occur without the other.

For continuous De Morgan triplets $\langle T, S, N\rangle$ it follows immediately that the negator has to be the standard negator.

Theorem 1 [9] *Consider a continuous De Morgan triplet $\langle T, S, N\rangle$. If $C_F(f)(\mathbf{x}) - D_F(f)(\mathbf{x})$ is independent of the Boolean function f, then N is the standard negator.*

To obtain the 'converse' implication that $N = \mathcal{N}$ forces the De Morgan property on T and S, we will first explore the system of functional equations $(2) = (3) = (4) = (5)$ more profoundly for general continuous triplets.

Lemma 1 [9] *Consider a continuous triplet (T, S, N). If $C_F(f)(\mathbf{x}) - D_F(f)(\mathbf{x})$ is independent of the Boolean function f, then there exist two I- automorphisms ϕ and ψ such that $T = (T_{\mathbf{L}})_\phi$, $S = (S_{\mathbf{L}})_\psi$ and $N = \mathcal{N}_\phi = \mathcal{N}_\psi$.*

Due Lemma 1 we are now able to compute the explicit forms of expressions (2)–(5):

$$(\psi^{-1}[\psi(y) - \psi(x)])^N - (\phi^{-1}[\phi(x) + \phi(y)])^N\,, \tag{6}$$

$$\psi^{-1}[\psi(x) + \psi(y)] - \phi^{-1}[\phi(y) - \phi(x)]\,, \tag{7}$$

$$\phi^{-1}[\phi(\psi^{-1}[\psi(x) + \psi(y)]) - \phi(\psi^{-1}[\psi(y) - \psi(x)])]\,, \tag{8}$$

$$1 - \psi^{-1}[1 + \psi(\phi^{-1}[\phi(y) - \phi(x)]) - \psi(\phi^{-1}[\phi(x) + \phi(y)])]\,, \tag{9}$$

At this point it remains unclear how to straightforwardly solve this system of functional equations. Both I-automorphisms (ϕ and ψ) and their inverses occur there each time in a different composition. To overcome this problem, we try to reduce the number of (unknown) I-automorphisms from two to one. The functional equation (8) = (9) can be transformed into

$$\gamma^{-1}[\gamma(x)+\gamma(y)]-\gamma^{-1}[\gamma(y)-\gamma(x)]=1-2\,\gamma^{-1}\left[\frac{1+\gamma(y-x)-\gamma(y+x)}{2}\right], \qquad (10)$$

for any $(x,y)\in[0,1/2]^2$, $x\leq y$ and with $\gamma: I\to I: x\mapsto \gamma(x)=\psi(\phi^{-1}[x])$ [9]. Unfortunately this procedure cannot be applied to the other equations of the system (6)=(7)=(8)=(9). Exploring the definition of γ more carefully, we can derive the following properties.

Proposition 1 [9] *Consider the continuous triplet (T,S,N) from Lemma 1 and let γ be the I-automorphism as defined above. Then the following observations hold:*

1. *γ is reciprocal:*
$$(\forall x\in I)(\gamma(1-x)=1-\gamma(x))\,,$$
2. *N is the standard negator if and only if it holds that*
$$(\forall x\in[0,1/2])(\gamma(2\,x)=2\,\gamma(x))\,.$$

We already showed that the De Morgan property implies $N=\mathcal{N}$. At this point we are finally able to prove that the converse property is also true. It is sufficient to show that in case $N=\mathcal{N}$, the I-automorphism γ is the identity mapping (Lemma 1).

Theorem 2 *Every reciprocal I-automorphism θ, fulfilling $\theta(2\,x)=2\,\theta(x)$ for every $x\in[0,1/2]$, must be the identity mapping.*

Combining Proposition 1 and Theorem 2 leads to the following result.

Theorem 3 [9] *Consider a continuous triplet (T,S,N) with $N=\mathcal{N}$. If $C_F(f)(\mathbf{x})-D_F(f)(\mathbf{x})$ is independent of the Boolean function f, then (T,S,N) is a De Morgan triplet.*

For continuous De Morgan triplets $\langle T,S,N\rangle$, with $N=\mathcal{N}$, we are able to characterize the solutions of (2) = (3) = (4) = (5). The Cauchy equation [1, 2] plays a key role in the reasoning.

Theorem 4 *Consider a continuous De Morgan triplet $\langle T,S,N\rangle$ with $N=\mathcal{N}$. Then (2) = (3) = (4) = (5) if and only if (T,S,N) is the Łukasiewicz triplet $\langle T_{\mathbf{L}},S_{\mathbf{L}},\mathcal{N}\rangle$.*

On the other hand we know that the Łukasiewicz triplet makes $C_F(f)(\mathbf{x})-D_F(f)(\mathbf{x})$ independent of the Boolean function f.

Theorem 5 [8] *For the Łukasiewicz triplet it holds that:*

$$C_F(f)(\mathbf{x}) - D_F(f)(\mathbf{x}) = 1 - \sum_{\mathbf{e}\in B^n} T(\mathbf{x}^{\mathbf{e}})\,,$$

for every n-ary Boolean function f and all $\mathbf{x} \in I^n$.

If a continuous triplet (T, S, N) fulfills the system of functional equalities (1) the De Morgan property is equivalent with $N = \mathcal{N}$ (Theorem 1 and Theorem 3). Therefore Theorem 4 can be rewritten as one of the following theorems.

Theorem 6 [9] *Consider a continuous triplet (T, S, N) with $N = \mathcal{N}$. Then $C_F(f)(\mathbf{x}) - D_F(f)(\mathbf{x})$ is independent of the Boolean function f if and only if (T, S, N) is the Łukasiewicz triplet $\langle T_{\mathbf{L}}, S_{\mathbf{L}}, \mathcal{N}\rangle$.*

Theorem 7 [9] *Consider a continuous De Morgan triplet $\langle T, S, N\rangle$. Then $C_F(f)(\mathbf{x}) - D_F(f)(\mathbf{x})$ is independent of the Boolean function f if and only if (T, S, N) is the Łukasiewicz triplet $\langle T_{\mathbf{L}}, S_{\mathbf{L}}, \mathcal{N}\rangle$.*

4 Conclusions

When working with the Łukasiewicz triplet, the difference between the conjunctive and disjunctive fuzzified normal form is independent of the Boolean function f. We have proven, by solving a system of functional equations, that the Łukasiewicz triplet is the only continuous De Morgan triplet for which this independence holds. Throughout the solution procedure, we have also laid bare the tight connection between the De Morgan law and N being the standard negator.

Acknowledgements

This work has been supported in part by the Bilateral Scientific and Technological Cooperation Flanders–Hungary BIL00/51 and by the EU COST Action 274 (TARSKI: Theory and Applications of Relational Structures as Knowledge Instruments).

References

[1] J. Aczél, *Diamonds are not Cauchy extentions best friend*, C.R. Math. Rep. Acad. Sci. Canada **5** (1983), 259–264.

[2] J. Aczél, *A Short Course on Functional Equations*, Theory and Decision Library; Series B: Mathematical and Statistical Methods, D. Reidel Publishing Company, 1987.

[3] T. Bilgiç, *Interval-valued preference structures*, European J. of Operational Research **105** (1998), 162–183.

[4] B. De Baets and J. Fodor, *Twenty years of fuzzy preference structures (1978-1997)*, Belgian Journal of Operations Research, Statistics and Computer Science **37** (1997), 61–82.

[5] J. Fodor and M. Roubens, *Fuzzy Preference Modelling and Multicriteria Decision Support*, Kluwer Academic Publishers, 1994.

[6] M. Gehrke, C. Walker, and E. Walker, *Normal forms and truth tables for fuzzy logics*, Fuzzy Sets and Systems **138** (2003), 25–51.

[7] E.P. Klement, R. Mesiar, and E. Pap, *Triangular norms*, Trends in Logic, Vol. 8, Kluwer Academic Publishers, 2000.

[8] K. Maes and B. De Baets, *Facts and figures on fuzzified normal forms*, IEEE Trans. on Fuzzy Systems (2005), to appear.

[9] K. Maes, B. De Baets and J. Fodor, *The unique role of the Łukasiewicz-triplet in the theory of fuzzified normal forms*, submitted.

[10] I. Perfilieva, *Normal forms for fuzzy logic functions and their approximation ability*, Fuzzy Sets and Systems **124** (2001), 371–384.

[11] I. Perfilieva, *Logical approximation*, Soft Computing **7** (2002), 73–78.

[12] I. Perfilieva, *Normal forms in BL-algebra and their contribution to universal approximation functions*, Fuzzy Sets and Systems **143** (2004), 111–127.

[13] I. Perfilieva and V. Kreinovich, *A new universal approximation result for fuzzy systems, which reflects CNF-DNF duality*, Internat. J. of Intelligent Systems **17** (2002), 1121–1130.

[14] I.B. Türkşen, *Interval-valued fuzzy sets based on normal forms*, Fuzzy Sets and Systems **20** (1986), 191–210.

[15] I.B. Türkşen, *Interval-valued fuzzy sets and 'compensatory AND'*, Fuzzy Sets and Systems **51** (1992), 295–307.

[16] I.B. Türkşen, *Fuzzy normal forms*, Fuzzy Sets and Systems **69** (1995), 319–346.

[17] I.B. Türkşen, *Non-specificity and interval-valued fuzzy sets*, Fuzzy Sets and Systems **80** (1996), 87–100.

[18] C. Walker and E. Walker, *Inequalities in De Morgan systems I*, Proceedings of the 2002 IEEE World Congress, Hawaii, 2002, pp. 607–609.

[19] C. Walker and E. Walker, *Inequalities in De Morgan systems II*, Proceedings of the 2002 IEEE World Congress, Hawaii, 2002, pp. 610–615.

k-Lipschitz t-norms

A. Mesiarová
Mathematical Institute of SAS
Štefánikova 49
81473 Bratislava, Slovakia
email: mesiarova@mat.savba.sk

Abstract

Inspired by an open problem of Alsina, Frank and Schweizer, k-Lipschitz t-norms and t-conorms are studied. Additive generators of k-Lipschitz t-norms are completely characterized.

Keywords: additive generator, k-Lipschitz property, triangular norm

1 Introduction

Triangular norms are, on the one hand, special semigroups and, on the other hand, solutions of some functional equations [1, 3, 6, 8]. This mixture quite often requires new approaches to answer questions about nature of triangular norms.

A triangular norm (t-norm for short) $T : [0,1]^2 \to [0,1]$ is an associative, commutative, non-decreasing function such that 1 acts as a neutral element [6]. Most important t-norms are the minimum $T_{\mathbf{M}}$, the product $T_{\mathbf{P}}$ and the Łukasiewicz t-norm $T_{\mathbf{L}}$ given by $T_{\mathbf{L}}(x,y) = \max(x+y-1,0)$. Observe that each continuous Archimedean t-norm T can be represented by means of a continuous additive generator [3, 4], i.e., a strictly decreasing continuous function $t : [0,1] \to [0,\infty]$ with $t(1) = 0$ such that

$$T(x,y) = t^{(-1)}(t(x) + t(y)),$$

where the pseudo-inverse $t^{(-1)} : [0,\infty] \to [0,1]$ in this special case is given by

$$t^{(-1)}(u) = t^{-1}(\min(u, t(0))).$$

Note that if t is an additive generator of a t-norm T then for any $d \in]0,\infty[$ also $d \cdot t$ is an additive generator of the t-norm T. For continuous t-norms also the opposite is true.

For the sake of completeness recall that each continuous t-norm (see [3, 4]) can be represented as an ordinal sum of continuous Archimedean t-norms (t-norm is called Archimedean if for each $(x, y) \in]0, 1[^2$ there is an $n \in \mathbb{N}$ with $x_T^{(n)} < y$, where $x_T^{(n)} = T(x, x_T^{(n-1)})$ and $x_T^{(1)} = x$). More precisely, for each continuous t-norm T there exists a unique (finite or countably infinite) index set A, a family of unique pairwise disjoint open subintervals $(]a_\alpha, e_\alpha[)_{\alpha\in A}$ and a family of unique continuous Archimedean t-norms $(T_\alpha)_{\alpha\in A}$ such that for all $(x, y) \in [0, 1]^2$

$$T(x,y) = \begin{cases} a_\alpha + (e_\alpha - a_\alpha) \cdot T_\alpha(\frac{x-a_\alpha}{e_\alpha - a_\alpha}, \frac{y-a_\alpha}{e_\alpha - a_\alpha}) & \text{if } (x,y) \in [a_\alpha, e_\alpha]^2, \\ \min(x,y) & \text{otherwise.} \end{cases}$$

We shall also write $T = (< a_\alpha, e_\alpha, T_\alpha >)_{\alpha\in A}$.

In the center of our interest are t-norms which satisfy k-Lipschitz property.

Definition 1
Let $T : [0, 1]^2 \to [0, 1]$ be a t-norm and let $k \in]0, \infty[$ be a constant. Then T is *k-Lipschitz* if

$$|T(x_1, y_1) - T(x_2, y_2)| \leq k \cdot (|x_1 - x_2| \ + \ |y_1 - y_2|) \tag{1}$$

for all $x_1, x_2, y_1, y_2 \in [0, 1]$.

Because of the neutral element $e = 1$, a t-norm can be k-Lipschitz only for $k \geq 1$. It is evident that if a t-norm T is k-Lipschitz it is also m-Lipschitz for any $m \in \mathbb{R}$, $k \leq m$. As it was shown in [5, 8], 1-Lipschitz t-norms are exactly those t-norms which are also copulas. By [5, 7], a strictly decreasing function $t : [0, 1] \to [0, \infty]$ with $t(1) = 0$ is an additive generator of a 1-Lipschitz Archimedean t-norm if and only if it is convex. The aim of this work is to give an answer to the open problem no. 11 from [2], i.e., to characterize and discuss k-Lipschitz t-norms.

Note that a partial answer to the problem of Alsina et al. posed in [2] was given by Y.-H. Shyu [9] who has showed that if the additive generator t of a t-norm T is differentiable and $t'(x) < 0$ for $0 < x < 1$, then T is k-Lipschitz if and only if $t'(y) \geq kt'(x)$ whenever $0 < x < y < 1$. This special case, as well as the characterization of additive generators of 1-Lipschitz t-norms, follow from our characterization.

2 k-Lipschitz t-norms and additive generators

Let T be a k-Lipschitz t-norm. Since it is k-Lipschitz it is evident that it is necessarily also continuous and it can be uniquely expressed as an ordinal sum of continuous Archimedean t-norms (for more details see [4]) which are then necessarily k-Lipschitz Archimedean t-norms. Furthermore, each of these k-Lipschitz Archimedean t-norm has a continuous additive generator.

Definition 2
Let $t : [0,1] \to [0,\infty]$ be a continuous strictly monotone function and let $k > 0$ be a real constant. Then t will be called *k-convex* if

$$t(x + k\epsilon) - t(x) \leq t(y + \epsilon) - t(y) \tag{2}$$

holds for all $x \in [0,1[\,, y \in]0,1[\,, \epsilon \in]0,1[$ where $x \leq y$ and $\epsilon \leq \min(1 - y, \frac{1-x}{k})$.

Note that because of the monotonicity of a continuous strictly decreasing function t can by k-convex only for $k \geq 1$. Moreover, when t is k-convex it is l-convex for all $l \geq k$. In the case of strictly increasing function, a continuous strictly increasing function c can by k-convex only for $k \leq 1$. Moreover, when c is k-convex it is l-convex for all $l \leq k$.

The following is an equivalent definition of k-convexity.

Lemma 1
Let $t : [0,1] \to [0,\infty]$ be a continuous strictly monotone function then the followings are equivalent.

(i) t is k-convex.

(ii) For all $x \in [0,1[\,, y \in]0,1[\,, \epsilon \in]0,1[$ where $x \leq y$ and $\epsilon \leq 1 - y$ it holds

$$t(\min(x + k\epsilon, 1)) - t(x) \leq t(y + \epsilon) - t(y). \tag{3}$$

Theorem 1
Let $T : [0,1]^2 \to [0,1]$ be an Archimedean t-norm and let $t : [0,1] \to [0,\infty]$ be an additive generator of T. Then T is k-Lipschitz if and only if t is k-convex.

Note that for $k = 1$ we get $t(y + \epsilon) - t(y) \geq t(x + \epsilon) - t(x)$ whenever $x \leq y$, $0 < \epsilon \leq 1 - y$, i.e., the function t is convex.

Corollary 1
Let $t : [0,1] \to [0,\infty]$ be an additive generator of a k-Lipschitz Archimedean t-norm. Let $x_0, y_0 \in]0,1], x_0 \leq y_0$ ($x_0, y_0 \in [0,1[, x_0 \leq y_0$). Then if there exist left (right) derivatives $t'_-(x_0)$ and $t'_-(y_0)$ ($t'_+(x_0)$ and $t'_+(y_0)$) we have

$$t'_-(x_0) \leq \frac{1}{k} t'_-(y_0)$$

$$(t'_+(x_0) \leq \frac{1}{k} t'_+(y_0)).$$

Moreover, let $z_0 \in]0,1[$ be such that both left and right derivatives $t'_-(z_0)$ and $t'_+(z_0)$ exist. Then we have

$$t'_-(z_0) \leq \frac{1}{k} t'_+(z_0).$$

From Corollary 1 and Theorem 1 follows the necessity of the result of Y.-H.Shyu [9]

Corollary 2 (Y.-H. Shyu)
Let $t : [0,1] \to [0,\infty]$ be an additive generator of a t-norm T, differentiable on $]0,1[$ and let $t'(x) < 0$ for $0 < x < 1$. Then T is k-Lipschitz if and only if $t'(y) \geq kt'(x)$ whenever $0 < x < y < 1$.

Corollary 3
Let $T : [0,1]^2 \to [0,1]$ be a continuous Archimedean t-norm and let $t : [0,1] \to [0,\infty]$ be an additive generator of T such that t is differentiable on $]0,1[\setminus \mathcal{R}$, where $\mathcal{R} \subset [0,1]$ is a set of isolated points. Then T is k-Lipschitz if and only if $kt'(x) \leq t'(y)$ for all $x, y \in [0,1]$, $x \leq y$ such that $t'(x)$ and $t'(y)$ exist.

Corollary 4
Let $t : [0,1] \to [0,\infty]$ be a strictly decreasing function differentiable on $]0,1[$. If $k \max\limits_{x\in[0,1]} t'(x) \leq \min\limits_{x\in[0,1]} t'(x)$ then t is an additive generator of some k-Lipschitz t-norm.

Example 1
(i) Let $t : [0,1] \to [0,\infty]$ be given by $t(x) = \frac{sin(\frac{\pi}{3}(1-x))}{\frac{\pi}{3}}$. Then $\max\limits_{x\in[0,1]} t'(x) = -\frac{1}{2}$ and $\min\limits_{x\in[0,1]} t'(x) = -1$, and hence $2 \max\limits_{x\in[0,1]} t'(x) \leq \min\limits_{x\in[0,1]} t'(x)$, i.e., t is an additive generator of some 2-Lipschitz t-norm.

(ii) Let $t : [0,1] \to [0,\infty]$ be given by $t(x) = (1-x) + \frac{(1-x)^2}{4}$. Then $\max\limits_{x\in[0,1]} t'(x) = -1$ and $\min\limits_{x\in[0,1]} t'(x) = -\frac{3}{2}$, and we have $\frac{3}{2} \max\limits_{x\in[0,1]} t'(x) \leq \min\limits_{x\in[0,1]} t'(x)$, i.e., t is an additive generator of some $\frac{3}{2}$-Lipschitz t-norm.

Although in the case of 1-Lipschitz t-norms their additive generators have left (right) derivative everywhere on $]0,1[$ (since $t(x) - t(x-\epsilon)$ is increasing when ϵ is decreasing), in the case of k-Lipschitz t-norms with $k > 1$ the situation is different. The following is an example of an additive generator of a 2-Lipschitz t-norm with no right derivative in $\frac{1}{2}$.

Example 2
Let $(a_n)_{n\in\mathbb{N}_0}$ be a sequence, where $a_n = \frac{1}{2} + (\frac{1}{2})^{n+1}$. Let $t : [0,1] \to [0,\infty]$ be given by

$$t(x) = \begin{cases} -x + \frac{1}{6}\frac{1}{2^{n+1}} + \frac{11}{12} & \text{if } x \in [a_{2n}, a_{2n+1}], n \in \mathbb{N}_0 \\ -\frac{x}{2} + \frac{2}{3} - \frac{1}{3}\frac{1}{2^{2n+3}} & \text{if } x \in]a_{2n+1}, a_{2n+2}], n \in \mathbb{N}_0 \\ -x + \frac{11}{12} & \text{if } x \in \left[0, \frac{1}{2}\right[. \end{cases}$$

Then t has no right derivative in point $\frac{1}{2}$. Moreover, since for all $x \in [0,1]$ and all $\epsilon \in]0, 1-x[$ it is $t(x+\epsilon) - t(x) \in [-\epsilon, -\frac{\epsilon}{2}]$ we have $t(x+2\epsilon) - t(x) \leq -\epsilon \leq t(y+\epsilon) - t(y)$ for all $x, y, 0 < \epsilon \leq \min(1-y, \frac{1-x}{2})$, i.e., t is 2-convex and due to Theorem 1 it is an additive generator of some 2-Lipschitz t-norm.

Note only that each continuous monotone function has derivative almost everywhere, i.e., the Lebesgue measure of the set S of all points from $[0,1]$ where derivative does not exist is equal to zero.

The following example shows that the requirement in Corollary 3 for points from $\mathcal{R}$ to be isolated is substantial.

Example 3
Let $t : [0,1] \to [0,\infty]$ be given by $t(x) = 1 - x + f(1-x)$ for all $x \in [0,1]$, where $f : [0,1] \to [0,1]$ is the Cantor function, i.e., $f(\frac{1}{3}) = f(\frac{2}{3}) = \frac{1}{2}$, etc. Then $t'(x) = -1$ for all $x \in [0,1]$ where $t'(x)$ exist. Since t is continuous and strictly decreasing with $t(1) = 0$ we know that t is an additive generator of some continuous t-norm. But t is not k-Lipschitz for any $k \in [1, \infty[$. For example

$$T(\frac{55}{81}, \frac{74}{81}) = t^{(-1)}(t(\frac{55}{81}) + t(\frac{74}{81})) = t^{-1}(\frac{26}{81} + \frac{7}{16} + \frac{7}{81} + \frac{3}{16}) = \frac{16}{27} + \frac{7}{8}$$

and

$$T(\frac{2}{3},\frac{74}{81}) = t^{(-1)}(t(\frac{2}{3})+t(\frac{74}{81})) = t^{-1}(\frac{1}{3}+\frac{1}{2}+\frac{7}{81}+\frac{3}{16}) = \frac{47}{81}+\frac{13}{16}.$$

We get that

$$T(\frac{55}{81},\frac{74}{81}) - T(\frac{2}{3},\frac{74}{81}) = \frac{1}{81}+\frac{1}{16} = \frac{97}{1296} = \frac{6.0625}{81}$$

and

$$\frac{55}{81}-\frac{2}{3} = \frac{1}{81}.$$

We have

$$|T(\frac{55}{81},\frac{74}{81}) - T(\frac{2}{3},\frac{74}{81})| > 6|\frac{55}{81}-\frac{2}{3}|,$$

i.e., T is not 6-Lipschitz. Similarly we can show for any $k \in [1,\infty[$ that T is not k-Lipschitz.

We will now continue in the investigation of additive generators of k-Lipschitz t-norms.

Proposition 1
Let $t : [0,1] \to [0,\infty]$ be an additive generator of a k-Lipschitz t-norm T. Then for any $x, y \in [0,1], x < y$ and any $z \in [x,y]$ we have

$$t(z) \leq \frac{z(kt(x)-t(y))+xt(y)-kyt(x)}{(k-1)z+x-ky}.$$

Remark 1
Supposing the differentiability of t on $[0,1]$, from Lagrange formula we get that $t(y)-t(\alpha x+(1-\alpha)y) = t'(\theta)(\alpha y-\alpha x)$ for some $\theta \in [\alpha x+(1-\alpha)y, y]$ and that $t(\alpha x+(1-\alpha)y)-t(x) = t'(\varphi)((1-\alpha)y-(1-\alpha)x)$ for some $\varphi \in [x, \alpha x+(1-\alpha)y]$. Since $\varphi \leq \theta$ from Corollary 2 we have $t'(\theta) \geq kt'(\varphi)$. We get

$$t(y)-t(\alpha x+(1-\alpha)y) \geq \frac{\alpha k}{1-\alpha}\left(t(\alpha x+(1-\alpha)y)-t(x)\right),$$

i.e.,

$$(1-\alpha)t(y)+\alpha k t(x) \geq (\alpha k+1-\alpha)t(\alpha x+(1-\alpha)y). \tag{4}$$

Recall the classical definition of convexity of a function t, in which for all $x, y \in \mathrm{Dom}(t)$ and $\alpha \in [0,1]$ it holds

$$t(\alpha x+(1-\alpha)y) \leq \alpha t(x)+(1-\alpha)t(y).$$

However, the last inequality is just the inequality (4) for $k = 1$.

Since the left derivative of the function $t(z) = \frac{k(z-1)}{z(k-1)-k}, z \in [0,1]$ in the point 0 is $t'(0) = -\frac{1}{k}$ and the right derivative in the point 1 is $t'(1) = -k$, from Corollary 2 it follows that this function is not itself an additive generator of some k-Lipschitz t-norm, but it is an additive generator of some k^2-Lipschitz t-norm. This means that the set of all normed additive generators of nilpotent k-Lipschitz t-norms has no strongest element and its supremum is the function $\frac{k(z-1)}{z(k-1)-k}$.

Corollary 5
Let $t : [0,1] \to [0,\infty]$ be an additive generator of a k-Lipschitz t-norm T. Then for any $x, y \in [0,1], x \leq y$ and any $z \in [x,y]$ we have

$$t(z) \leq t(x) + \frac{1}{k}\frac{t(x)-t(y)}{x-y}(z-x)$$

and

$$t(z) \leq t(y) + k\frac{t(x)-t(y)}{x-y}(z-y)$$

Acknowledgement This work was supported by grants APVT 20-046402 and VEGA 2/3163/23. A partial support of the international project COST 274 and CEEPUS SK-42 is also kindly announced.

References

[1] J. Aczél (1966). *Lectures on Functional Equations and their Applications.* Academic Press, New York.

[2] C. Alsina, M. J. Frank, B. Schweizer (2003). Problems on associative functions. *Aequationes Math.* **66**, pp. 128–140.

[3] E. P. Klement, R. Mesiar, E. Pap, (2000). *Triangular Norms.* volume 8 of Trends in Logic, Studia Logica Library, Kluwer Acad. Publishers, Dordrecht.

[4] C. M. Ling (1965). Representation of associative functions. *Publ. Math. Debrecen*, **12**, pp. 189–212.

[5] R. Moynihan (1978). On τ_T semigroups of probability distribution functions II. *Aeqiationes Math.*, **17**, pp. 19–40.

[6] B. Schweizer and A. Sklar (1960). Statistical metric spaces. *PACJM*, **10**, pp. 313–334.

[7] B. Schweizer, A. Sklar (1963). Associative functions and abstract semigroups. *Publ. Math. Debrecen*, **10**, pp. 69–81.

[8] B. Schweizer, A. Sklar (1983). *Probabilistic Metric Spaces.* North-Holland, New York.

[9] Y.-H. Shyu (1984). *Absolute continuity in the τ_T-operations.* PhD Thesis, Illinois Institute of Technology, Chicago.

Conditionally distributivity of a uni-norm with respect to a triangular conorm

Dragan Jočić

Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad
Trg D. Obradovića 4, 21000 Novi Sad, Serbia and Montenegro
e-mail:drjocic@PTT.yu

1. Introduction

Many different kinds of operation defined on subset of real numbers play fundamental roles in many important fields as for exmple in fuzzy set theory, fazzy logic, neural nets, operation research, optimization problems, differential equations etc. Special intention is paid to operations defined on interval of reals. The examples are t-norms and t-conorms which act on the interval [0,1], pseudo-additions and pseudo-multiplications in the sense of Sugeno and Murofushi [9] which act on the interval $[0,\infty]$ or in the sense of E. Pap [6] which act on the interval [a,b] where is [a,b] is closed subinterval of $[-\infty, +\infty]$, compesatory operators (Klement, Mesiar, Pap [5]) and uninorms (Fodor, Yager, Rybalov [1]).

In this paper we will consider a binary operations on unit interval i.e. above mentioned t-norms, t-conorms and uninorms.In many situations those operations have to be distributive one with respect to other as for example in the context of integrals based on decomposable measures. If the range of pseudo-aditive measures is subset of the unit interval [0,1], countinous t-conorms S are natural candidates for the pseudo-adition, leading to the concept of S-decomposable measures. For generalized Lebesque integral for [0,1]-valued functions, a second operation U turning ([0,1], S,U) into a semiring will be considered. Consequently U should be comutative, associative, non-decreasing, and should have a neutral element e from (0,1] i.e. it should be a uninorm (if $e \in (0,1)$) or a t-norm (if $e = 1$). Also some distributivity of U over S (so-called conditional distributivity) will be required (so [0,1], S, U) becomes a conditionally distributive semiring.

2. Preliminaries

Triangular norms and conorms were originally introduced in the context of probablistic metric spaces. A triangular norm (shortly t-norm) i a binary operation on the unit interval which is commutative, associative, non-decreasing in each component and which has 1 as a neutral element. Dually, a triangular conorm (shortly t-conorm) is a binariy operation on the unit interval which is commutative ,associative, non-decreasing in each component, and which has 0 as a neutral element. . The most important t-norms are the minimum T_M , the product T_P, and the Lukasiewicz t-norm T_L, which (together with the corresponding t-conorms maximum S_M, probablistic sum S_P, and Lukasiewicz t-conorm S_L) are given by

$T_M(x,y) = \min(x,y)$, $S_M(x,y) = \max(x,y)$
$T_P(x,y) = xy$ $S_P(x,y) = x+y-xy$
$T_L(x,y) = \max(x+y-1,0)$ $S_L(x,y) = \min(x+y,1)$

Each countinuous Archimedian t-norm T has a multiplicative generator i.e. a countinnuous, strictly increasing function $\Theta:[0,1]\rightarrow[0,1]$ satisfying $\theta(1)=1$ such that $T(x,y)=\Theta^{(-1)}(\Theta(x)\Theta(y))$ where $\Theta^{(-1)} : [0,1]\rightarrow[0,1]$ is the pseudo-inverse of Θ given by $\Theta^{(-1)}(x) = \Theta^{-1}(\min(x,\Theta(1))$

The countinuous, strictly increasing functions $s : [0,1]\rightarrow[0,\infty]$ satisfying $s(0)=0$ serve as aditive generators of countinuous Archimedean t-conorms S as follows: $S(x,y)=s^{(-1)}(s(x)+s(y))$. In particular , S is strict if and only if $s(1)=\infty$ (i.e. if s is bijection) and S is nilpotent if and only if $s(1)<\infty$.

Each continuous t-norm(t-conorm) can be represented as an ordinal sum of continuous Archimedean t-norms(t-conorms) i.e. there exists a uniquely determined (finite or coutable infinite) index set A, a family of uniquely determined pairwise disjoint open subintervals $(a_\alpha, b_\alpha)_{\alpha\in A}$ of [0,1] and a family of uniquely determined continuous Archimedean t-norms(t-conorms) $(T_\alpha)_{\alpha\in A}$ such that $T=(<a_\alpha,e_\alpha,T_\alpha>)_{\alpha\in A}$ where each $<a_\alpha,e_\alpha,T_\alpha>$ is called summand.

A third class of operations will be important for us the so-called uninorms. Uninorrms are generalizations of t-norms and t-conorms allowing the neutral element lying anywhere in the unit interval [0,1]. Therefore a uninorm is binary operation on the unit interval which is commutative, associative, non-decreasing in each component and which has a neutral element e from [0,1]. Suppose U is a uninorm with neutral element e from (0,1). Define two functions T_U and S_U on the unit square as follows

$$T_U(x,y) = \frac{U(ex,ey)}{e} \quad x,y\in[0,1] \tag{1}$$

$$S_U(x,y)=\frac{U(e+1-e)x,e+(1-e)y)-e}{1-e} \quad x,y\in[0,1] \tag{2}$$

It is easy to verifay that T_U defined by (1) is a t-norm and S_U defined by (2) is a t-conorm. Therefore the structure of uninorms on the squares $[0,e]^2$ and $[e,1]^2$ is closely related to t-norms and t-conorms. That is we have

$$U(x,y)=eT\left(\frac{x}{e},\frac{y}{e}\right) \quad if\ 0\le x,y\le e \tag{3}$$

and

$$U(x,y)=e+(1-e)S\left(\frac{x-e}{1-e},\frac{y-e}{1-e}\right) \quad if\ e\le x,y\le 1 \tag{4}$$

with some t-norm T i some t-conorm S. T is called the underlying t-norm of U and S is called the underlying t-conorm of U.

Concerning the definicion of U on the rest of unit square we have that

$$\min(x,y)\le U(x,y)\le \max(x,y) \text{ if } x\le e\le y \text{ or } x\ge e\ge y$$

Each increasing bijection f:[0,1]→[0,∞] defines (using the convention $0\cdot\infty=0$) a left-countinuous uninorm U (there exist no continuous uninorm) with neutral element $f^{-1}(1)$ by means of $U(x,y)=f^{-1}(f(x)f(y))$.

The generators of t-conorms, t-norms and uninorms suggest that a t-conorm can be seen as transformations of the addition of non-negative real numbers, wheres uninorms and t-norms are transformations of multiplications.

Throughout this paper we shall work with a continuous t-conorm S and left-contninuous uninorm U satisfying the following conditional distributivity (CD)

$U(x,S(y,z))=s(U(x,y),U(x,z))$ for all x,y,z from [0,1] with $S(y,z)<1$ (CD)

In this context we shall refer to ([0,1],S,U) as a conditionally distributive semiring

3. Conditional distributivity

In this section we will consider two cases depending on neutral element e of uninorm U. The first case is when e=1 and then U becomes a t-torm T. The second case is when $e\in(0,1)$.

3.1 Conditional distributivity of t-norm T over t-conorm S

The first case is described in the following theorem whose proof can be found in [3]

Theorem 1 *A continuous t-norm T and countinuous t-conorm S satisfies the condition (CD) if and only if we have either one of the following cases*

(i) $S=S_M$

(ii) There is a strict t-norm T^* and a nilpontent t-conorm S^* such that the additive generator s of S^* satisfaying s(1)=1 is also a multiplicative generator of T^*, and there is an a $\in [0,1[$ such that for some countinuous t-norm T^{**}, we have $T=(<o,a,T^{**}>,<a,1,T^*>)$ and $S=(<a,1,S^*>)$.

Remark 1 If in the functional equation (CD) we omit the condition $S(y,z)<1$ we say that T is distributive over S and then we have only trivial solutions, i.e., $S=S_M$.

This remark shows how much the distributivity lows restrict the choice of possible t-conorms. Thus it seems reasonable to restrict the domain of the distributivity low if we look for solutions which are not trivial.

A full characterization of all pairs (T,S) satisfying the condition (CD) which are not continuous is still an open problem.

3.2 Conditional distributivity of uninorm U over t-conorm S

In this subsection we give a characterization of all pairs (U,S) satisfying (CD) where U is a left-continuous uninorm with neutral element $e\in(0,1)$ and S is a continuous t-conorm. In this subsection we will distinguish two cases. The first is when neutral element e of the uninorm U is an idempotent element of the t-conorm S. The second case is when neutral element e of the uninorm U is not an idempotent element of the t-conorm S.

Theorem 2 *A left-continuous uninorm U with neutral element $e\in(0,1)$ and a continuous t-conorm S where e is an idempotent element of S satisfy (CD) if and only if $S=S_{M..}$*

The second case is more complicated and in order to investigate it we shall prove a sequence of lemmas. Firstly we present a lemma in which the ordinal sum structure for a continuous t-conorm simplifies considerablly.

Lemma 1 *Let U be a left-continuous uninorm with neutral element $e\in(0,1)$ and let S be a continuous t-conorm for which e is not an idempotent element. If the pair (U,S) satisfies the condition (CD), then $|A|=1$.*

Lemma 2 *Let U be a left-continuous uninorm with neutral element $e\in(0,1)$ and S be a continuous t-conorm for which e is not an idempotent element . If the pair (U,S)*

satisfies the condition (CD), then U(x,y)∈ $[a,b]$ for all x,y ∈ $[a,b]$, where a, b are from the previous Lemma 1 such that S=(<a,b,S>).*

So far we have seen that when (CD) is satisfied then ordinal sum representation for t-conorm S is simplfied because we have only one summand <a,b,S*>. Also we have showed that U(x,y)∈ $[a,b]$ when x,y∈ $[a,b]$, i.e., uninorm U is compatibile with structure of t-conorm S.

Now we can apply results from [4]

Theorem 3 *Let ($[0,1]$, U, S) be a conditionally distributive semiring*

(i) If S is strict t-conorm, i.e if it is generated by a bijective additive generator s:[0,1]→[0,∞], then U is generated by c·s for some constant c from (0,∞) and hence has the neutral element $s^{-1}(\frac{1}{c})$

(ii) If S is a nilpotent t-conorm, i.e., if it has a (unique) additive generator s which can be seen as an increasingbijection s:[0,1]→[0,1], then U is a t-norm with multiplicative generator s.

References

[1] J. C. Fodor, R. R. Yager, A. Rybalov, Structure of Uninorms, Int. J. of Uncertainty, Fuzziness and Knowledge-based Systems 5 N.4 (1997) 411-427.

[2] J. S. Golan, The Theory of Semirings with Applications in mathematics and Teoretical Computer Science, Longman, Essex 1992.

[3] E. P. Klement, R. Mesiar, E. Pap, Triangular Norms, volume 8 of Trends in Logic, Studia Logica Library, Kluwer Academic Publishers, Dordrecht, 2000.

[4] E. P. Klement, R.Mesiar, E. Pap, Integration with respect to decomposable measures,based on a conditionally distributive semirings on the unit interval, Int. J. Uncertainty Fuzziness Knowledge-Based Systems 8 (2000) 701-717

[5] E. P. Klement, R. Mesiar, E. Pap, On the relationship of associative compesatory operators to triangular norms and conorms, Int. J. Uncertainty Fuzziness Knowledge-Based Systems 4 (1996) 129-144

[6] E. Pap, Null-Additive Set Functions, Kluwer Academic Publishers, Dordrecht (1995)

[7] E. Pap, N. Ralević, Pseudo-operations on finite intervals, Novi Sad J.Math. Vol.29,No. 1 (1999) 9-18

[8] B. Schweizer, A. Sklar, Probablistic Metric Spaces, Elsevier-North Holland New York (1983).

[9] M. Sugeno, T. Murofushi, Pseudo-additive measures and integrals, J. Math. Anal. Appl. 122 (1987) 197-222.

Aggregation of Reciprocal Relations

Susanne Saminger
Johannes Kepler Universität
susanne.saminger@jku.at

Koen Maes
Ghent University
koen.maes@Ugent.be

Johannes Kepler Universität Linz

Institut für Wissensbasierte Mathematische Systeme
Johannes Kepler Universität Linz
A-4040 Linz
Austria

Fuzzy Logic Laboratorium Linz-Hagenberg

Softwarepark Hagenberg
Hauptstrasse 99
A-4232 Hagenberg
Austria

1 Introduction

In many areas of decision theory the question arises how to determine a collective decision, preference, or opinion based on several individual decisions, preferences or opinions. Different techniques can be applied for achieving that goal. One of the possible strategies is simply to carry out an aggregation process based on the experts' decisions. This is usually done by some aggregation operator which maps arbitrarily but countably many input values to one output value. Inputs and outputs belong to the same domain and the output should be representative for the input data itself or at least for some of its aspects.

We will focus on fuzzy preference relations, more precisely on reciprocal relations and their aggregation. Reciprocal relations are binary fuzzy relations R fulfilling $R(a,b) + R(b,a) = 1$ and are known under various names such as ipsodual relations, probabilistic relations, ... [1]. In preference modelling such relations can be used to render the individual intensity of preference. Consider a finite set of alternatives $\{x_1, \ldots, x_m\}$ and n experts, $n \in \mathbb{N}$. The opinion of expert k is represented by the fuzzy preference relation R_k, such that $R_k(x_i, x_j)$ expresses the degree by which expert k prefers alternative x_i over alternative x_j. Sometimes $R_k(x_i, x_j)$ is abbreviated by the notation r_{ij}^k (see, e.g., [2, 3, 4]).

In order to avoid inconsistent preferences it is often required that the degree to which x_i is preferred to x_j is in some sense complementary to the degree to which x_j is preferred to x_i. The latter can be obtained by using reciprocal preference relations R, i.e., $R(x_i, x_j) + R(x_j, x_i) = 1$. In this case two alternatives x_i and x_j are indifferent if $R(x_i, x_j) = R(x_j, x_i) = \frac{1}{2}$. Note that in general, a crisp preference structure (P, I) can be associated with each reciprocal relation R which is based on its strict α-cuts with $\alpha \in \left[\frac{1}{2}, 1\right[$ (see e.g. [2]).

2 Self-dual aggregation operators

Our aim is to aggregate reciprocal relations into another reciprocal relation by means of an aggregation operator.

Definition 1 ([5]). *A function* $\mathbf{A} : \bigcup_{n\in\mathbb{N}}[0,1]^n \to [0,1]$ *is called an* aggregation operator *if it fulfills the following properties*

(AO1) $\mathbf{A}(x_1, \ldots, x_n) \leq \mathbf{A}(y_1, \ldots, y_n)$ *whenever* $x_i \leq y_i$ *for all* $i \in \{1, \ldots, n\}$, $n \in \mathbb{N}$,

(AO2) $\mathbf{A}(x) = x$ *for all* $x \in [0,1]$,

(AO3) $\mathbf{A}(0, \ldots, 0) = 0$ *and* $\mathbf{A}(1, \ldots, 1) = 1$.

Each aggregation operator $\mathbf{A}$ can be represented by a family $(\mathbf{A}_{(n)})_{n\in\mathbb{N}}$ of n-ary operations, i.e. functions $\mathbf{A}_{(n)} : [0,1]^n \to [0,1]$ given by

$$\mathbf{A}_{(n)}(x_1, \ldots, x_n) = \mathbf{A}(x_1, \ldots, x_n).$$

Note that , $\mathbf{A}_{(1)} = \mathrm{id}_{[0,1]}$ and, for $n \geq 2$, each $\mathbf{A}_{(n)}$ is non-decreasing and satisfies the boundary conditions $\mathbf{A}_{(n)}(0, \ldots, 0) = 0$ and $\mathbf{A}_{(n)}(1, \ldots, 1) = 1$. Usually, an aggregation operator $\mathbf{A}$ and

its corresponding family $(\mathbf{A}_{(n)})_{n\in\mathbb{N}}$ of n-ary operations are identified with each other. The n-ary operations $\mathbf{A}_{(n)} : [0,1]^n \to [0,1]$, $n \geq 2$ are also referred to as *n-ary aggregation operators.*

An aggregation operator can also be defined to act on any closed interval $I = [a,b] \subseteq [-\infty,\infty]$. Then we will speak of an *aggregation operator acting on I*. While **(AO1)** and **(AO2)** basically remain the same, only the boundary condition **(AO3)** has to be modified accordingly

(AO3') $\mathbf{A}(a,\ldots,a) = a$ and $\mathbf{A}(b,\ldots,b) = b$.

However, since we will use aggregation operators to combine reciprocal relations it is appropriate to consider only aggregation operators with respect to the unit interval. Considering arbitrary reciprocal relations $R_i : X \times X \to [0,1]$, $i \in \{1,\ldots,n\}$, $n \in \mathbb{N}$, on some universe X, we want to investigate whether the aggregated relation R on X defined by

$$R(a,b) = \mathbf{A}(R_1(a,b),\ldots,R_n(a,b)) \tag{1}$$

for some aggregation operator $\mathbf{A}$ is again reciprocal, i.e., fulfilling $R(a,b) + R(b,a) = 1$ for all $a,b \in X$. It can be shown (and has already been mentioned in [7]) that an aggregation operator $\mathbf{A}$ preserves the reciprocity of the involved relations if and only if it is self-dual, i.e.,

$$\mathbf{A}(x_1,\ldots,x_n) = 1 - \mathbf{A}(1-x_1,\ldots,1-x_n)$$

for all $x_i \in [0,1]$, $i \in \{1,\ldots,n\}$. Note that in the literature several other names have been used for expressing the self-duality, e.g., neutrality [3] and reciprocity [2, 7].

Examples of self-dual aggregation operators are the arithmetic mean, all weighted arithmetic means, $\mathrm{med}_{0.5}$, quasi-arithmetic means with $f(1-x) = 1 - f(x)$, OWA operators for which the corresponding weighting triangle is self-reversed. Note that no t-norm, t-conorm, or uninorm is self-dual.

In general, self-dual operations have already been characterized and investigated by Silvert [8] in 1978, denoting such operations as "symmetric sums". In [5], this characterization has been applied to aggregation operators.

Proposition 2 ([5]). *An aggregation operator $\mathbf{A}$ is self-dual if and only if there exists an aggregation operator $\mathbf{B}$ such that $\mathbf{A} = \mathbf{B}^\sharp$ where*

$$\mathbf{B}^\sharp(x_1,\ldots,x_n) = \frac{\mathbf{B}(x_1,\ldots,x_n)}{\mathbf{B}(x_1,\ldots,x_n) + \mathbf{B}(1-x_1,\ldots,1-x_n)} \tag{2}$$

with the convention $\frac{0}{0} = 0.5$.

If $\mathbf{A}$ is self-dual, it is sufficient to choose $\mathbf{B} = \mathbf{A}$ in order to obtain $\mathbf{A} = \mathbf{B}^\sharp$. However, in general several aggregation operators $\mathbf{B}$ are appropriate for modelling the same self-dual aggregation operator $\mathbf{A}$. Therefore, $\mathbf{B}$ is not uniquely determined and each set $\{\mathbf{B} \mid \mathbf{A} = \mathbf{B}^\sharp\}$, with $\mathbf{A}$ some aggregation operator, can be seen as an equivalence class on the set of all aggregation operators.

Besides this characterization, García-Lapresta and Marques Pereira have mentioned another characterization of self-dual aggregation operators which is based on the arithmetic mean.

Proposition 3 ([7]). *An aggregation operator* $\mathbf{A}$ *self-dual if and only if there exists an aggregation operator* $\mathbf{B}$ *such that*

$$\mathbf{A}(x_1,\ldots,x_n) = \frac{(\mathbf{B}(x_1,\ldots,x_n) + 1 - \mathbf{B}(1-x_1,\ldots,1-x_n))}{2}.$$

Both characterizations are, however, only examples of a more general characterizations method. Besides these algebraical characterizations, we were also able to introduce an approach for building self-dual aggregation operators based on a more geometrical point of view.

3 Properties of aggregation operators

As an aggregation operator often fulfills some additional conditions depending on its field of application, we also explored the effect these extra conditions have on self-dual aggregation operators. We provide here only the different properties we have tackled.

Definition 4. *Consider some aggregation operator* $\mathbf{A} : \bigcup_{n\in\mathbb{N}} [0,1]^n \to [0,1]$.

(i) $\mathbf{A}$ *is called* symmetric*, if*

$$\forall n \in \mathbb{N}, \forall x_1,\ldots,x_n \in [0,1] : \mathbf{A}(x_1,\ldots,x_n) = \mathbf{A}(x_{\alpha(1)},\ldots,x_{\alpha(n)})$$

for all permutations $\alpha = (\alpha(1),\ldots,\alpha(n))$ *of* $\{1,\ldots,n\}$.

(ii) $\mathbf{A}$ *is called* associative *if*

$$\begin{aligned}&\forall n,m \in \mathbb{N}, \forall x_1,\ldots,x_n,y_1,\ldots,y_m \in [0,1] :\\ &\qquad \mathbf{A}(x_1,\ldots,x_n,y_1,\ldots,y_m) = \mathbf{A}(\mathbf{A}(x_1,\ldots,x_n),\mathbf{A}(y_1,\ldots,y_m)).\end{aligned}$$

for arbitrarily many inputs

(iii) $\mathbf{A}$ *is called* idempotent *if*

$$\forall n \in \mathbb{N}, \forall x \in [0,1] : \mathbf{A}(\overbrace{x,\ldots,x}^{n \text{ times}}) = x.$$

(iv) An element $e \in [0,1]$ *is called a neutral element of* $\mathbf{A}$ *if*

$$\begin{aligned}&\forall n \in \mathbb{N}, \forall x_1,\ldots,x_n \in [0,1] :\\ &\quad \text{if } x_i = e \text{ for some } i \in \{1,\ldots,n\} : \mathbf{A}(x_1,\ldots,x_n) = \mathbf{A}(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n).\end{aligned}$$

(v) An element $a \in [0,1]$ *is called an* annihilator *of* $\mathbf{A}$ *if*

$$\forall n \in \mathbb{N}, \forall x_1,\ldots,x_n \in [0,1] : \text{if } x_i = a \text{ for some } i \in \{1,\ldots,n\} : \mathbf{A}(x_1,\ldots,x_n) = a.$$

(vi) $\mathbf{A}$ *is called* continuous*, if*

$$\forall n \in \mathbb{N} : \mathbf{A}_{(n)} \text{ is continuous}.$$

(vii) $\mathbf{A}$ *fulfills the* Lipschitz property *with constant* $c \in]0, \infty[$ *(is* c-Lipschitz *for short) if*

$$\forall n \in \mathbb{N}, \forall x_1, \ldots, x_n, y_1, \ldots, y_m \in [0,1]: \\ |\mathbf{A}(x_1, \ldots, x_n) - \mathbf{A}(y_1, \ldots, y_n)| \leq c \cdot \sum_{i=1}^{n} |x_i - y_i|.$$

Symmetric aggregation operators are also known as *commutative* aggregation operators. From multi-criteria decision making idempotency is sometimes also called *unanimity* since it expresses unanimity if all criteria involved in the decision making process are fulfilled to the same degree x. Examples of idempotent aggregation operators are the minimum and the maximum, but also the arithmetic mean. The product is a non-idempotent aggregation operator.

Continuity does not imply the c-Lipschitz property for some $c \in]0, \infty[$. Also observe that for different $n \in \mathbb{N}$ the Lipschitz constant c can vary and hence should be denoted by c_n. Because of the boundary condition **(AO3)** the smallest Lipschitz constant c_n related to $\mathbf{A}_{(n)}$ is $c_n = \frac{1}{n}$ and because of the condition **(AO2)** an aggregation operator $\mathbf{A}$ can never be c-Lipschitz with $c < 1$. The arithmetic mean is an example of a 1-Lipschitz aggregation operator and is the unique aggregation operator, such that $\mathbf{A}_{(n)}$ is $\frac{1}{n}$-Lipschitz. Further examples of 1-Lipschitz aggregation operators are the minimum, the product and the Łukasiewicz t-norm. Note that, any t-norm is a symmetric, associative aggregation operator with neutral element 1 and annihilator 0.

For most of the foregoing properties we were able to characterize the self-dual aggregation operators fulfilling the extra conditions in question covering the (so far) known results from the literature dealing with special subclasses of aggregation operators. Combining several of these conditions further restricts the set of acceptable self-dual aggregation operators.

4 Final Remarks

Having a look on the existing literature on the aggregation of reciprocal relations we have to state that there is real need to investigate that topic on a general level. The results mentioned so far are part of ongoing research related to the topic. Other results, e.g., when imposing some transitivity conditions on reciprocal relations, have already been achieved and will surely be further investigated by the authors in the future.

References

[1] J.-P. Doignon, B. Monjardet, M. Roubens, and Ph. Vincke. Biorder families, valued relations, and preference modelling. *J. Math. Psychol.*, 30:435–480, 1986.

[2] J.L. Garciá-Lapresta and B. Llamazares. Aggregation of fuzzy preferences: Some rules of the mean. *Soc.Choice Welfare*, 17:673–690, 2000.

[3] J.L. Garciá-Lapresta and B. Llamazares. Majority decisions based on difference of votes. *Jorunal of Mathematical Economics*, 35:463–481, 2001.

[4] F. Chiclana, F. Herrera, E. Herrera-Viedma, and L. Martínez. A note on the reciprocity in the aggregation of fuzzy preference relations using owa operators. *Fuzzy Sets and Systems*, 137:71–83, 2003.

[5] T. Calvo, A. Kolesárová, M. Komorníková, and R. Mesiar. Aggregation operators: properties, classes and construction methods. volume 97 of *Studies in Fuzziness and Soft Computing*, pages 3–104. Physica-Verlag, Heidelberg, 2002.

[6] E. P. Klement, R. Mesiar, and E. Pap. *Triangular Norms*, volume 8 of *Trends in Logic. Studia Logica Library*. Kluwer Academic Publishers, Dordrecht, 2000.

[7] J. L. Garciá-Lapresta and R.A. Marques Pereira. Constructing reciprocal and stable aggregation operators. In *Proceedings AGOP '2003, Alcalá de Henares*, pages 73–78, 2003.

[8] W. Silvert. Symmetric summation: A class of operations on fuzzy sets. *IEEE Trans. Systems Man Cybernet.*, 9:657–659, 1979.

[9] M. J. Frank. On the simultaneous associativity of $F(x, y)$ and $x + y - F(x, y)$. *Aequationes Math.*, 19:194–226, 1979.

[10] A. Kolesárová. 1-Lipschitz aggregation operators and quasi-copulas. *Kybernetika (Prague)*, 39:615–629, 2003.

Knowledge Based Methods in Management

Thomas Biringer
Magna Steyr Fahrzeugtechnik
e-mail *thomas.biringer@magnasteyr.com*

Abstract — In the car production industry the time to market is in the future a key indicator for the ability to solve a lot of money in reducing the time to market. In this case it is necessary to use new methods to solve the obtained knowledge from running projects in a knowledge database for further projects. In this case I will show a possible way to secure obtained knowledge with using cybernetic model and machine learning.

Key words — *Knowledge, cybernetic model, machine learning*

Johannes Kepler Universität Linz

Institut für Algebra, Stochastik und
wissensbasierte mathematische Systeme
Johannes Kepler Universität Linz
A-4040 Linz
Austria

Fuzzy Logic Laboratorium Linz-Hagenberg

FLLL
Softwarepark Hagenberg
Hauptstrasse 99
A-4232 Hagenberg
Austria

1 Introduction

Reffering to the topic ”start-up-management” there are no essays which ones demonstrate the whole process incipient with development up to start of production, inclucing all involved divisions. There are no fruitful disquisitions beyond the extensive cohesions among development, logistics, serialproduction und their interact. So there is no basis for controlling a startup and for rating the existing risk to meet the deadline, achieve quality and expenses. Presently there is no conclusive knowledge base for a start of production. Its interesting to regard the start up concerning ”novel product, new craft, new employees” from the point of view of the producing plant. To my mind its witted - on the basis of the experience of the successful start up - to create a knowledge base for future startups and modeling the start up with new techniques for knowledge engineering and to compare the results of modeling to the values of the real start up.

Within the start up it became clear, that - due to the huge complexness of the organisational structure - the present ”Technomorphe Managementsystem” is on upper bound respectively the networked processes cant be acquired conclusive and the parameters as a whole dont flow into decision making. Due to the nonlinear mutual processinteractions and the non-controllable complexity there often arises a limited consideration for problems, whereby the management is pushed into a reactive part respectively in worst case, it elects for a non-target-oriented measure. Due to the ever-shortend development-period it comes as well to a intensive integration of processes. Thus its essential to find the primal factors of success to control the start of production. To find the essential factors (canals with huge information content), there should be applied new treatments from different fields of knowledge (such as Maschinenlernen) and be checked on their capability. In ”Technomorphen Systemen”, where its assumed, that enough information is available, its applied at best in clear systems, so its needful to arrange a ”Systemisches Modell”, what it used as base for controllability ”Maschinenlernen”. In line with ”Maschinenlernen” its essential to use a algorithm, what isnt just adopt to anticipate information, but to bring out a linguistic context from observed facts out of the basic start-up. By this means it gets possible to operators, to discover abnormal cohesions out of the basic start-up and if necessary taking corrective action.

2 Use of Machine Learning Methods

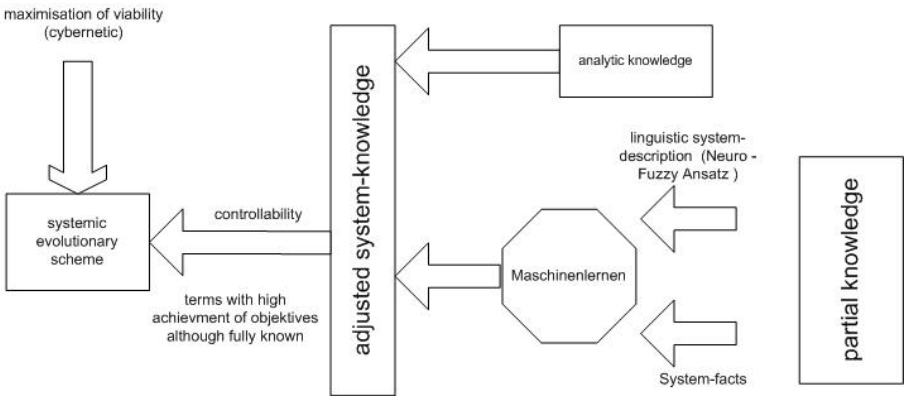


Figure 1: Sytemisches Modell

Approach of machine learning without linguistic description

$$T = \{(x_1, y_1), (x_2, y_2), ... (x_L, y_L)\} \tag{1}$$

Sought after : Funktion f with

$$\widehat{f}(x) = \widehat{y} \approx y \tag{2}$$

Target: Good prediction

$$G = \{(x_1, y_1), (x_2, y_2), ... (x_M, y_M)\} \tag{3}$$

Because of the enhancement of machine learning about the linguistic description the system knowledge is expanded from pure data analyze with interpretable knowledge.

$$\widehat{f}(x) = \widehat{y} \approx y \text{ with } \widehat{y} \approx y + \text{ interpretation of y}$$

Thereby you are able to calculate the inversion of this problem. Therefore it is possible to modify the input data because of the knowledge of the required initial state.

$$\widehat{f}^{-1}(y) = x \; with \; \widehat{y} \approx y + \; interpretation \; of \; y \tag{4}$$

For this purpose it is necessary to describe the impact data and parameters from the system with exact linguistic data.

3 Complex lunch structure

Description of all input data in linguistic way. For handling this problem an approach of a hierarchical controller design will be used.

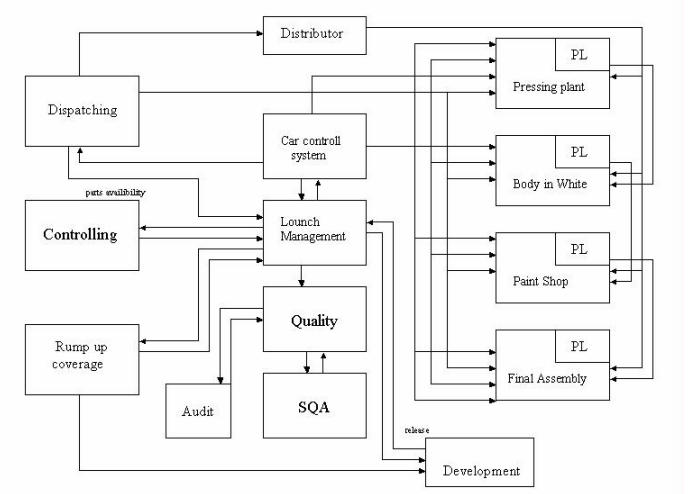


Figure 2: Complex System

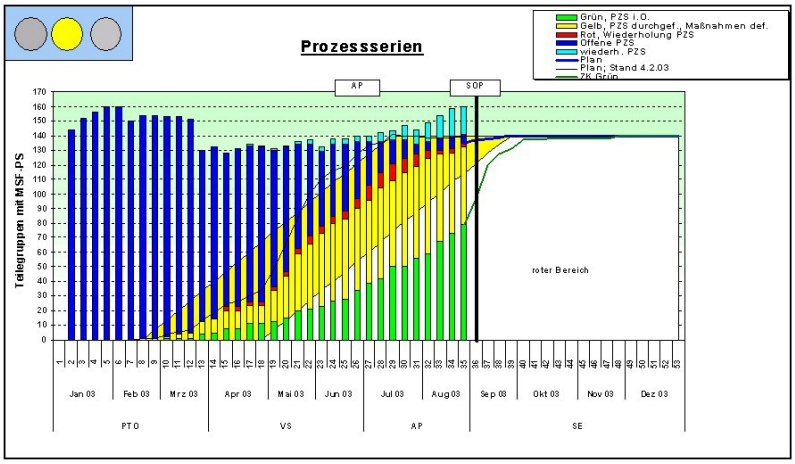


Figure 3: Part Readiness

The knowledge is available in empiric data.(See Figure 3)

For the linguistic description the triangular symmetrical membership funktions is applicated.

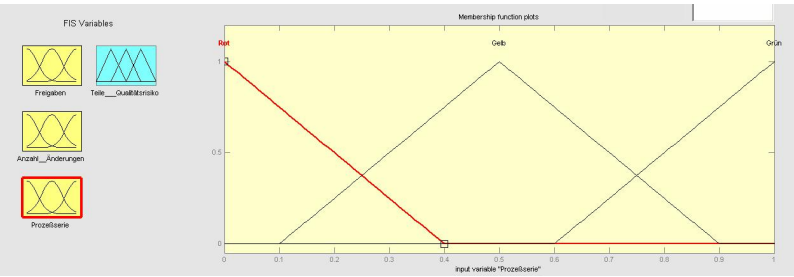


Figure 4: Part Quality

3.1 MISO system (multiple Input/single output)

Given: A conventional MISO system (See Figure 6), performing the mapping as follows:[7](Page 66 ff)

$$y = (x_1, x_2, x_3, x_4, ...x_n) \tag{5}$$

An input vector **X** is defined in the Cartasian product space of the universe of discourse of particular inputs $X_1 X_2 ... X_n$.

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_n \end{bmatrix}$$

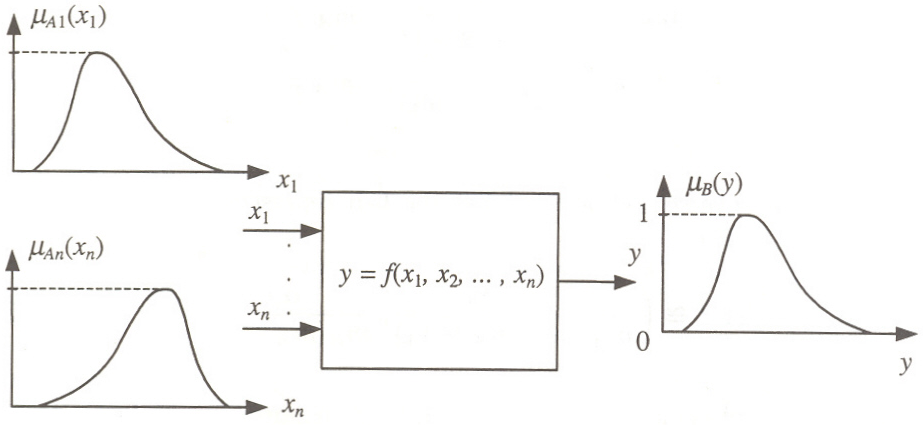


Figure 5: MISO System

The function f maps the set elements belonging to the universe of discourse of the input vector **X** onto the universe of discourse of the output *Y*.

$$f : X_1 \times X_2 \times .... \times X_n \tag{6}$$

3.2 Mandani fuzzy modell

The Mandani fuzzy modell of the system can have the form od a set of rules as well as membership functions represented (See Figure 5)[7](Page 281 ff)

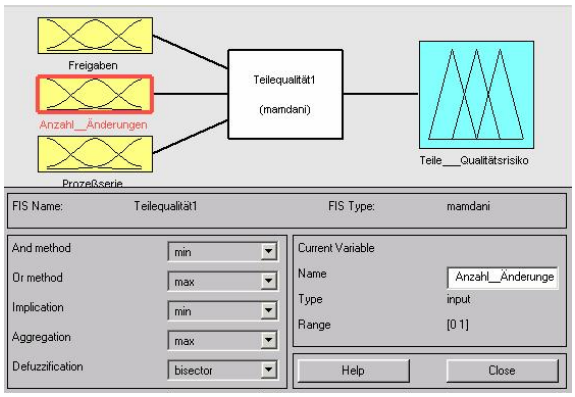


Figure 6: Fuzzy system

1: If [Freigaben is früh] and [Anzahl Änderungen is wenig] and [Prozessserie is gelb]then [Teile Qualitätsrisiko is gering]

References

[1] Fredmund MALIK: Systemisches Management Evolution, Selbstorganisation.

[2] Fredmund MALIK: Strategie des Managements komplexer Systeme.

[3] Ludwig WAGNER: Flexible Unternehmensstrukturen.

[4] Christian BORGELT, Frank KLAWONN: Neuro Fuzzy Syteme.

[5] K.Schlacher: Systemtheorie, Vorlesungsskriptum J.K. Universität Linz, Kap. 6.2, 1995.

[6] K.Schlacher: Moderne Frequenzbereichsmethoden, Vorlesungsskriptum J.K. Universität Linz, Kap. 2.3, 1996.

[7] Andrzej Piegat: Fuzzy Modeling and Control,

Technical Report
SCCH–TR–04xy

Real-time Computations at the Edge of Chaos and Self-Organized Criticality in Recurrent Neural Networks

Thomas Natschläger
Software Competence
Center Hagenberg
A-4232 Hagenberg, Austria
Thomas.Natschlaeger@scch.at

Nils Bertschinger
Max Planck Institute for
Mathematics in the Sciences
D-04103 Leipzig, Germany
bertschi@mis.mpg.de

Robert Legenstein
Institute for Theoretical
Computer Science, TU Graz
A-8010 Graz, Austria
legi@igi.tu-graz.ac.at

Abstract — The computational capabilites of randomly connected networks of threshold gates in the time-series domain are analyzed. In particular we propose a complexity measure which we find to assume its highest values near the edge of chaos, i.e. the transition from ordered to chaotic dynamics. Furthermore we show that the proposed complexity measure predicts the computational capabilites very well: only near the edge of chaos are such networks able to perform complex computations on time series. Additionally a simple synaptic scaling rule for self-organized criticality is presented and analyzed.

Key words — *recurrent neural networks, chaotic dynamics, real-time computation, self-organised criticality*

Kplus
Kompetenzzentren-Programm

1 Introduction

It has been proposed that extensive computational capabilities are achieved by systems whose dynamics is neither chaotic nor ordered but somewhere in-between order and chaos. This has led to the idea of "*computation at the edge of chaos*". Early evidence for this hypothesis has been reported e.g. in [1]. The results of numerous computer simulations carried out in these studies suggested that there is a sharp transition between ordered and chaotic dynamics. Later on this was confirmed by Derrida and others [2]. They used ideas from statistical physics to develop an accurate mean-field theory which allowed to determine the critical parameters analytically. Because of the physical background this theory focused on the autonomous dynamics of the system, i.e. its relaxation from an initial state (the input) to some terminal state (the output) without any external influences. In contrast to such "offline" computations we will focus in this article on time-series computations, i.e. mappings, also called filters, from a time-varying input signal to a time-varying output signal. Such "online" or real-time computations describe more adequately the input to output relation of systems like animals or autonomous robots which must react in real-time to a continously changing stream of sensory input.

The purpose of this paper is to analyze how the computational capabilities of randomly connected recurrent neural networks in the domain of real-time processing and the type of dynamics induced by the underlying distribution of synaptic weights are related to each other. In particular we will show that for the types of neural networks considered in this paper (defined in Sec. 2) there also exists a transition from ordered to chaotic dynamics. This transition from ordered to chaotic dynamics is determined using an extension of the mean-field approach described in [3] and [4] (Sec. 3). As the next step we propose a novel complexity measure (Sec. 4) which can be calculated using the mean-field theory developed in Sec. 3 and serves as a predictor for the computational capability of a network in the time series domain. Employing a recently developed framework for analyzing real-time computations [5, 6] we investigate in Sec. 5 the relationship between network dynamics and the computational capabilities in the time-series domain. In Sec. 6 of this paper we propose and analyze a synaptic scaling rule for self-organized criticality for the types of networks considered here. In contrast to previous work [7] we do not only check that the proposed rule shows adaptation towards critical dynamics but also show that the computational capabilities of the network are actually increased if the rule is applied.

Relation to previous work: In [5] the so called *liquid state machine* (LSM) approach was proposed and used do analyze the computational capabilities in the time-series domain of randomly connected networks of biologically inspired network models (composed of leaky integrate-and-fire neurons). We will use that approach to demonstrate that only near the edge of chaos complex computations can be performed (see Sec. 5). A similar analysis for a restricted case (zero mean of synaptic weights) of the network model considered in this paper can be found in [4].

2 The Network Model and its Dynamics

We consider *input driven* recurrent networks consisting of N threshold gates with states $x_i \in \{0, 1\}$. Each node i receives nonzero incoming weights w_{ij} from exactly K randomly chosen nodes j. Each nonzero connection weight w_{ij} is randomly drawn from a Gaussian distribution with mean μ and variance σ^2. Furthermore the network is *driven by an external input signal* $u_{(.)}$ which is injected into each node. Hence, in summary the update of the network state $\boldsymbol{x}_t = (x_{1,t}, \ldots, x_{N,t})$ is given by $x_{i,t} = \Theta\left(\sum_{j=1}^{N} w_{ij} \cdot x_{j,t-1} + u_{t-1}\right)$ which is applied for all neurons

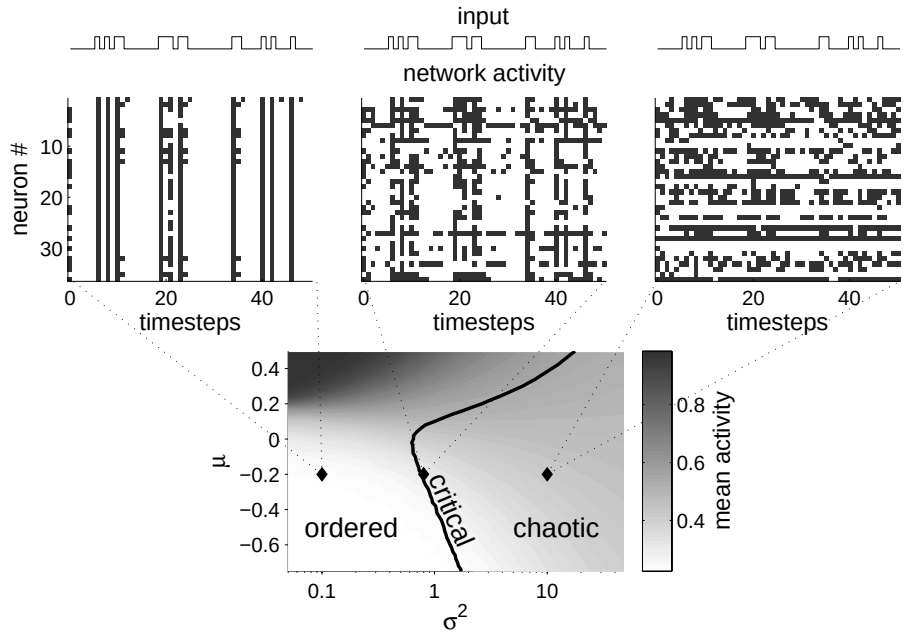


Figure 1: Networks of randomly connected threshold gates can exhibit ordered, critical and chaotic dynamics. In the upper row examples of the time evolution of the network state x_t are shown (black: $x_{i,t} = +1$, white: $x_{i,t}) = 0$, input indicated above) for three different networks with parameters taken from the ordered, critical and chaotic regime respectively. Parameters: $K = 5$, $N = 500$, $\bar{u} = -0.5$, $r = 0.3$ and σ^2 and μ as indicated in the phase plot below. The background of the phase plot shows the mean activity a^* (see Sec. 3) of the networks depending on the parameters σ^2 and μ.

in parallel and where $\Theta(h) = 1$ if $h \geq 0$ and $\Theta(h) = 0$ otherwise. In the following we consider a randomly drawn binary input signal $u_{(.)}$: at each time step u_t assumes the value $\bar{u} + 1$ with probability r and the value $\bar{u}$ with probability $1 - r$. This network model is similar to the one we have considered in [4]. However it differs in two important aspects: a) By using states $x_i \in \{0, 1\}$ we emphasis the asymmetric information encoding by spikes prevalent in biological neural systems and b) it is more general in the sense that the Gaussian distribution from which the non-zero weights are drawn is allowed to have an arbitrary mean $\mu \in \mathbb{R}$. This implies that the network activity $a_t = \frac{1}{N} \sum_{i=1}^{N} x_{i,t}$ can vary considerably for different parameters (compare Fig. 1) and enters all the calculations discussed in the rest of the paper.

The top row of Fig. 1 shows typical examples of ordered, critical and chaotic dynamics (see the next section for a definition of order and chaos). The system parameters corresponding to each type of dynamics are indicated in the lower panel (*phase plot*). We refer to the (phase) transition from the ordered to the chaotic regime as the *critical line* (shown as the solid line in the phase plot). Note that increasing the variance σ^2 of the weights consistenctly leads to chaotic behavior.

3 The Critical Line: Order and Fading Memory versus Chaos

To define the chaotic and ordered phase of an *input driven* network we use an approach which is similar to that proposed by Derrida and Pomeau [2] for autonomous systems: consider two (initial) network states with a certain (normalized) Hamming distance. These states are mapped to their corresponding successor states (using the same weight matrix) *with the same input* in each case and the change in the Hamming distance is observed. If small distances tend to grow this is a sign of chaos whereas if the distance tends to decrease this is a signature of order.

Following closely the arguments in [4, 3] we developed a mean-field theory (see [8] for all details) which allows to calculate the update $d_{t+1} = f(d_t, a_t, u_t)$ of the normalized Hamming distance $|\{i : x_{i,t} \neq \tilde{\boldsymbol{x}}_{i,t}\}|/N$ between two states $\boldsymbol{x}_t$ and $\tilde{x}_t$ as well as the update $a_{t+1} = A(a_t, u_t)$ of the network activity in one time step. Note that d_{t+1} depends on the input u_t (in contrast to [3]) and also on the activity a_t (in contrast to [4]). Hence the two-dimensional map $F_u(d_t, a_t) := (d_{t+1}, a_{t+1}) = (f(d_t, a_t, u_t), A(a_t, u_t))$ describes the time evolution of d_t and a_t given the input times series $u_{(\cdot)}$.

Let us consider the stady state of the averaged Hamming distance f^* as well as the stady state of the averaged network activity, i.e. $(f^*, a^*) = \lim_{t\to\infty} \left\langle F_u^t(d, a)\right\rangle$.[1] If $f^* = 0$ we know that any state differences will eventually die out and the network is in the ordered phase. If on the other hand a small difference is amplified and never dies out we have $f^* \neq 0$ and the network is in the chaotic phase. Whether $f^* = 0$ or $f^* \neq 0$ can be decided by looking at the slope of the function $f(\cdot, \cdot, \cdot)$ at its fixed point $f^* = 0$. Since a_t does not depend on d_t we calculate the averaged steady state activity a^* and determine the slope α^* of the map $rf(d, a, \bar{u} + 1) + (1 - r)f(d, a, \bar{u})$ at the point $(d, a) = (0, a^*)$. Accordingly we say that the network is in the ordered, critical or chaotic regime if $\alpha^* < 1$, $\alpha^* = 1$ or $\alpha^* > 1$ respectively. In [8] it is shown that the so called critical line $|\alpha^*| = 1$ where the phase transition from ordered to chaotic behavior occurs is given by

$$P_{bf} = \sum_{n=0}^{K-1} \binom{K-1}{n} a^{*n}(1 - a^*)^n (rQ(1, n, \bar{u} + 1) + (1 - r)Q(1, n, \bar{u})) = \frac{1}{K} \qquad (1)$$

Where P_{bf} denotes the probability (averaged over the inputs and the network activity) that a node will change its output if a single out of its K input bits is flipped.[2] Examples of critical lines that were calculated from this formula (marked by the solid lines) can be seen in Fig. 2 for $K = 5$ and $K = 10$.[3]

The ordered phase can also be described by using the notion of *fading memory* (see [5] and the references therein). Intuitively speaking in a network with fading memory a state $\boldsymbol{x}_t$ is fully determined by a finite history $u_{t-T}, u_{t-T+1}, \ldots, u_{t-1}, u_t$ of the input $u_{(\cdot)}$. A slight reformulation of this property (see [6] and the references therein) shows that it is equivalent to the requirement that all state differences vanish, i.e. being in the ordered phase. Fading memory plays an important role in the "liquid state machine" framework [5] since together with the separation property (see below) it would in principle allow an appropriate *readout function* to deduce the recent input,

[1] F_u^t denotes t-fold composition of the map $F_u(\cdot, \cdot)$ where in the k-th iteration the input u_k is applied and $\langle\cdot\rangle$ denotes the average over all possible initial conditions and all input signals with a given statistics determined by $\bar{u}$ and r.

[2] The actual singe bit-flip probability Q depends on the number n of inputs which are 1 and the actual input and is given by $Q(1, n, u) = \int_{-\infty}^{-u} \phi(\xi, n\mu, n\sigma^2)\left(1 - \Phi(-u - \xi, \mu, \sigma^2)\right) d\xi + \int_{-u}^{\infty} \phi(\xi, n\mu, n\sigma^2)\Phi(-u - \xi, \mu, \sigma^2)d\xi$ and ϕ, Φ denote the Gaussian density and cumulative density respectively (see [8] for a detailed explanation).

[3] For each value of $\mu = -0.6 + k * 0.01$, $k = 0..100$ a search was conducted to find the value for σ^2 such that $\alpha^* = 1$. Numerical iterations of the map function A where used to determine a^*.

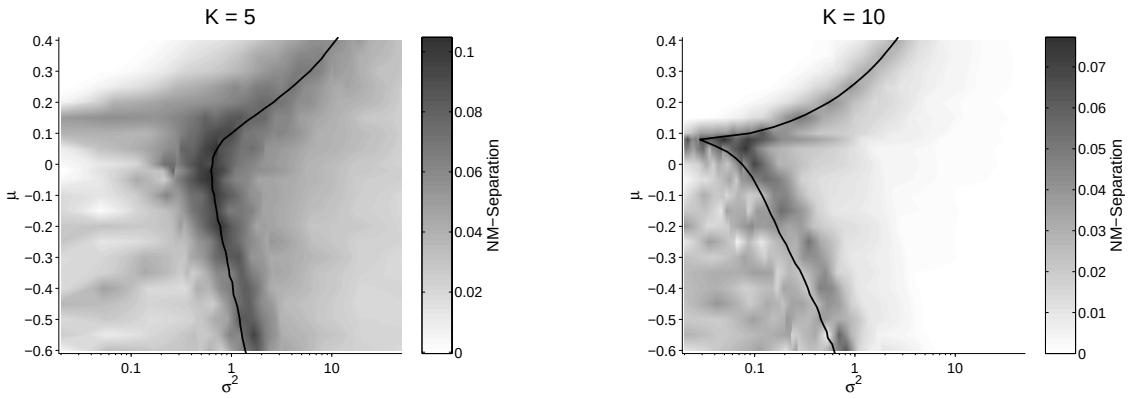


Figure 2: NM-separation assumes high values on the critical line. The gray coded image shows the NM-separation in dependence on σ^2 and μ for K denoted in the panels, $r = 0.3$, $\bar{u} = -0.5$ and $b = 0.1$. The solid line marks the critical values for σ^2 and μ.

or any function of it, from the network state. If on the other hand the network does not have fading memory (i.e. is in the chaotic regime) a given network state $x(t)$ also contains "spurious" information about the initial conditions and hence it is hard or even impossible to deduce any features of the recent input.

4 *NM*-Separation as a Predictor for Computational Power

As already mentioned the *separation property* [5] is especially important if a network is to be useful for computations on input time-series: only if different input signals separate the network state, i.e. different inputs result in different states, it is possible for a readout function to respond differently. Hence it is clear that for any two different input time series for which the readout function should produce different outputs should drive the recurrent network into two different states.

The mean field theory we have developed (see [8]) can be extended to describe the time evolution of the Hamming distance of the network states (i.e. the separation) and the network activities that result from applying different inputs $u_{(\cdot)}$ and $\tilde{u}_{(\cdot)}$ with a mean distance of $b := \Pr\{u_t \neq \tilde{u}_t\}$. In this case the three-dimensional map $S_{u,\tilde{u}}(d_t, a_t, \tilde{a}_t) := (d_{t+1}, a_{t+1}, \tilde{a}_{t+1}) = (s(d_t, a_t, \tilde{a}_t, u_t, \tilde{u}_t), A(a_t, u_t), A(\tilde{a}_t, \tilde{u}_t))$ fully describes the time evolution of the Hamming distance and the network activities. In the following we will denote by s^*, a^*, and $\tilde{a}^*$ the steady state of the averaged Hamming distance and averaged network activities, i.e. $(s^*, a^*, \tilde{a}^*) = \lim_{t\to\infty} \left\langle S^t_{u,\tilde{u}} \right\rangle$.

The overall separation for a given input statistics (determined by $\bar{u}$, r, and b) is given by s^*. However, this overall separation measurement can not be directly related to the computational power since chaotic networks separate even minor differences in the input to a very high degree. The part of this separation that is caused by the input distance b and not by the distance of some initial state is then given by $s^* - f^*$ because f^* measures the state distance that is caused by differences in the initial states and remains even after long runs with the same inputs (see Sec. 3). Note that f^* is always zero in the ordered phase and non-zero in the chaotic phase.

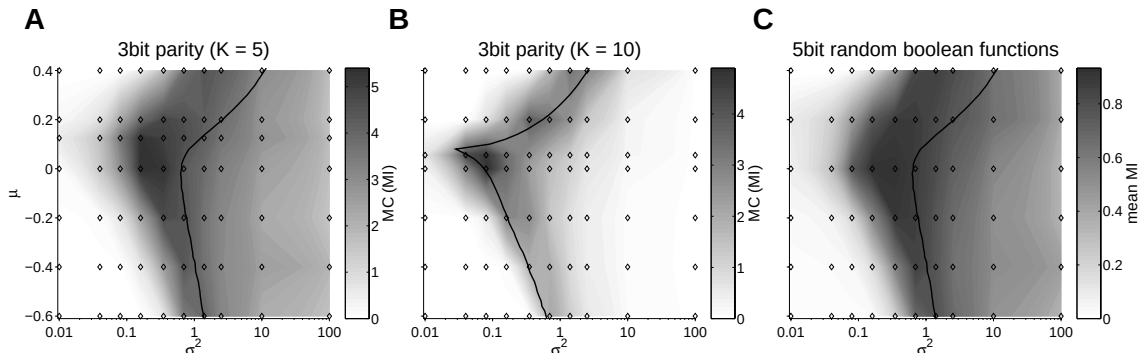


Figure 3: Real-time computation at the edge of chaos. **A** The gray coded image (an interpolation between the data points marked with open diamonds) shows the performance of trained networks in dependence of the parameters μ and σ^2 for the delayed 3-bit parity task. Performance is measured as the memory capacity $MC = \sum_\tau I(v, y^{(\tau)})$ where $I(v, y^{(\tau)})$ is the mutual information between the classifier output $v_{(\cdot)}$ and the target function $y_t^{(\tau)} = \mathrm{PARITY}(u_{t-\tau}, u_{t-\tau-1}, u_{t-\tau-2})$ measured on a test set. **B** Same as panel A but for $K = 10$. **C** Same as panel A but for an average over 50 randomly drawn Boolean functions f of the last 5 time steps, i.e. $y = f(u_t, u_{t-1}, ..., u_{t-4})$.

Since we want the NM-separation to be a predictor for computational power we correct $s^* - f^*$ by a term which takes the separation into account which is due to a predominant input drive. A suitable measure for this "immediate separation" i^* is the average *increase* in the Hamming distance i^* if the system is run for a long time ($t \to \infty$) with equal inputs $u_{(\cdot)} = \tilde{u}_{(\cdot)}$ and then a single step with an input pair $(v, \tilde{v})$ with an average difference of $b = \Pr\{v, \neq \tilde{v}\}$ is applied: $i^* = s(f*) - f* = \lim_{t\to\infty} \sum_{v,\tilde{v}=0}^{1} r^v (1-r)^{1-v} b^{|v-\tilde{v}|} (1-b)^{1-|v-\tilde{v}|} \left\langle s(F_u^t) \right\rangle - f^*$. Hence a measure of the network mediated separation NM_{sep} due to input differences is given by

$$NM_{sep} = s^* - f^* - i^* \tag{2}$$

In Fig. 2 the NM-separation resulting from an input difference of $b = 0.1$ is shown in dependence of the network parameters μ and σ^2.[4] Note that the NM-separation peaks at the critical line. Because of the computational importance of the separation property this also suggests that the computational capabilities of the networks will peak at the onset of chaos, which is confirmed in the next section.

5 Real-Time Computations at the Edge of Chaos

To access the computational power of a network we make use of the so called "liquid state machine" framework which was proposed by Maass et.al. [5] and indepenently by Jaeger [6]. They put forward the idea that any complex time-series computation can be implemented by composing a system which consists of two conceptually different parts: a) a properly chosen general-purpose

[4] For each value of $\mu = -0.6 + k * 0.05$, $k = 0..20$ 10 values for σ^2 where chosen near the critical line and 10 other values where equally spaced (on a logarithmic scale) over the interval [0.02,50]. For each such pair (μ, σ^2) extensive numerical iterations of the map S where performed to get accurate estimates for s^*, f^* and i^*.

recurrent network with "rich" dynamics and b) a readout function that is trained to map the network state to the desired outputs (see [5, 6, 4] for more details). This approach is potentially successful if the general-purpose network encodes the relevant features of the input signal in the network state in such a way that the readout function can easily extract it. We will show that near the critical line the networks considered in this paper encode the input in such a way that a simple linear classifier $C(\boldsymbol{x}(t)) = \Theta(\boldsymbol{w} \cdot \boldsymbol{x}(t) + w_0)$ suffices to implement a broad range of complex nonlinear filters. Note that in order to train the network for a given task only the parameters $\boldsymbol{w} \in \mathbb{R}^N$, $w_0 \in \mathbb{R}$ of the linear classifier are adjusted such that the actual network output $v(t)$ is as close as possible to the target values $y(t)$.

To access the computational power in a principled way networks with different parameters were tested on a delayed 3-bit parity task for increasing delays and on randomly drawn Boolen functions of the last 5 input bits. Note that this tasks are quite complex for the networks considered here since most of them are not linear separable (i.e. the parity function) and require memory. Hence to achieve good performance it is necessary that a state $\boldsymbol{x}_t$ contains information about several input bits $u_{t'}$, $t' < t$ in a nonlinear transformed form such that a linear classifier C is sufficient to perform the nonlinear computations.

The results are summarized in Fig. 3 where the performance (measured in terms of mutual information) on a test set between the network output and the target signal is shown for various parameter settings (for details see [4]). The highest performance is clearly achieved for parameter values close to the critical line were the phase transition occurs. This has been noted before [1]. In contrast to these previous results the networks used here are not optimized for any specific tasks but ther computational capabilities are assessed by is evaluating it for many different tasks. Therefore a network that is specifically designed for a single task will not show a good performance in this setup. These considerations suggest the following hypotheses regarding the computational function of generic recurrent neural circuits: to serve as general-purpose temporal integrator, and simultaneously as kernel (i.e., nonlinear projection into a higher dimensional space) to facilitate subsequent linear readout of information whenever it is needed.

6 Self-Organized Criticality via Synaptic Scaling

Since the computational capabilities of a network depend crucially on having almost critical dynamics an adaptive system should be able to adjust its dynamics accordingly.

Equ. (1) states that critical dynamics are achieved if the probability P_{bf} that a single bit-flip in the input shows up in the output should on average (over the external and internal input statistics given by $\bar{u}$, r and a^* respectively) be equal to $\frac{1}{K}$. To allow for a rule that can adjust the weights of each node a local estimate of P_{bf} must be available. This can be accomplished by estimating P_{bf} from the margin, i.e. the distance of the internal activation from the firing threshold, of each node. Intuitively a node with an activation that is much higher or lower than its firing threshold is rather unlikely to change its output if a single bit in its input is flipped. Formally P_{bf}^i of node i is given by the average (over the internal and external input statistics) of the following quantity:

$$\frac{1}{K} \sum_{j=1, w_{ij}>0}^{N} \Theta((w_{ij}(1 - 2x_j(t-1))(1 - 2x_i(t)) - m_i(t))) \tag{3}$$

where $m_i(t) = \left| \sum_{j=1}^{N} w_{ij} x_j(t-1) + u_t \right|$ denotes the margin of node i (see [8] for details). Each node now applies synaptic scaling to adjust itself towards the critical line. Accordingly we arrive

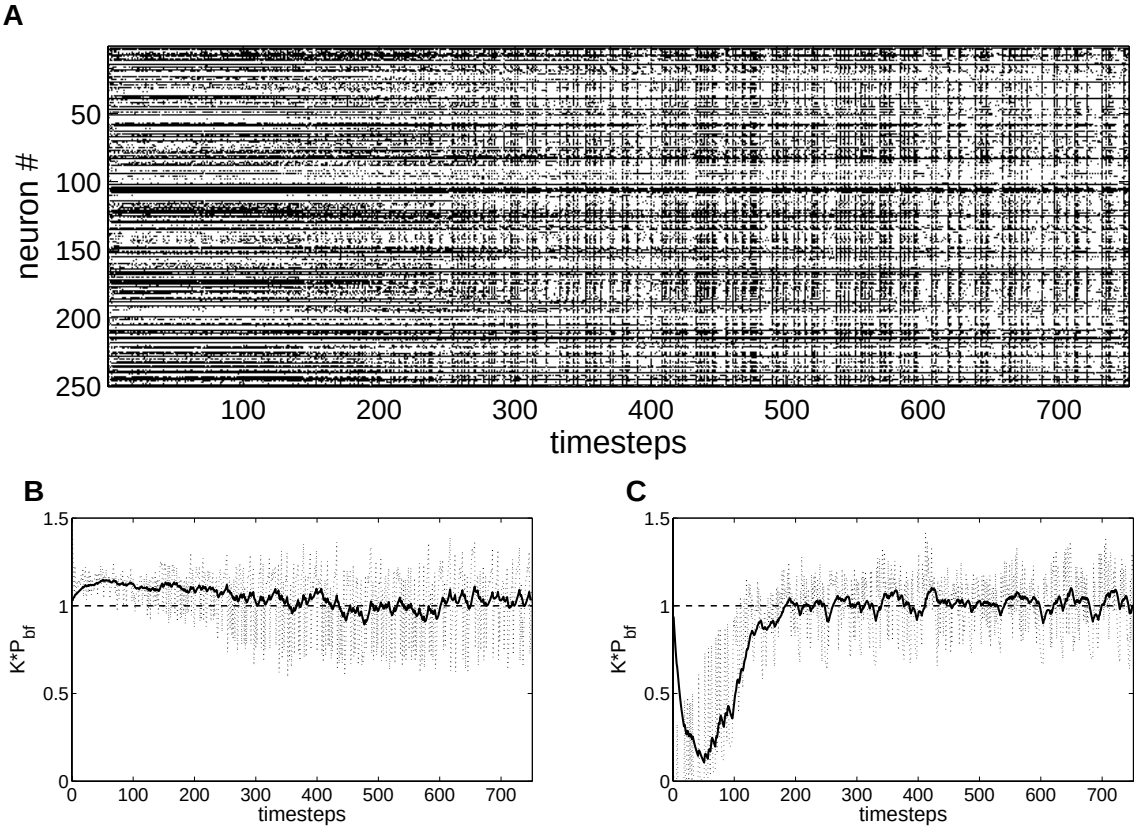


Figure 4: Self-organized criticality. **A** Time evolution of the network state $\boldsymbol{x}(t)$ starting in a chaotic regime while the rule (4) is active (black: $x_i(t) = +1$, white: $x_i(t) = 0$). Parameters: $K = 5, \bar{u} = -0.5, r = 0.3, \mu = 0$ and initial $\sigma^2 = 100$. **B** Estimated P_{bf}. The dotted line shows how the node averaged estimate of P_{bf} evolves over time for the network shown in A. The running average of this estimate (thick black line) as used by the rule (4) clearly shows that P_{bf} approaches its critical value (dashed line). **C** Same as B but for $K = 10$ and initial $\sigma^2 = 0.01$ in the ordered regime.

at the following rule:

$$w_{ij}(t+1) = \begin{cases} \frac{1}{1+\nu} \cdot w_{ij} & \text{if } P_{bf}^{est_i}(t) > \frac{1}{K} \\ (1+\nu) \cdot w_{ij}(t) & \text{if } P_{bf}^{est_i}(t) < \frac{1}{K} \end{cases} \tag{4}$$

where $\nu \ll 1$ is the learning rate and $P_{bf}^{est_i}(t)$ is a running average of the formula in Equ. (3) to estimate P_{bf}^{i}. Applying this rule in parallel to all nodes of the network is then able to adjust the network dynamics towards criticality as shown in Fig. 4. The upper row shows the time evolution of the network states $\boldsymbol{x}(t)$ while the SOC rule Equ. (4) is running. It is clearly visible how the network dynamics changes from chaotic (the initial network had the parameters $K = 5, \mu = 0$ and $\sigma^2 = 100$) to critical dynamics that respect the input signal. The lower row of Fig. 4 shows how the averaged estimated bitflip probability $\frac{1}{N}\sum_{i=1}^{N} P_{bf}^{est_i}(t)$ approaches its critical value for the case of the above network and one that started in the ordered regime ($K = 10, \mu = 0, \sigma^2 = 0.01$).

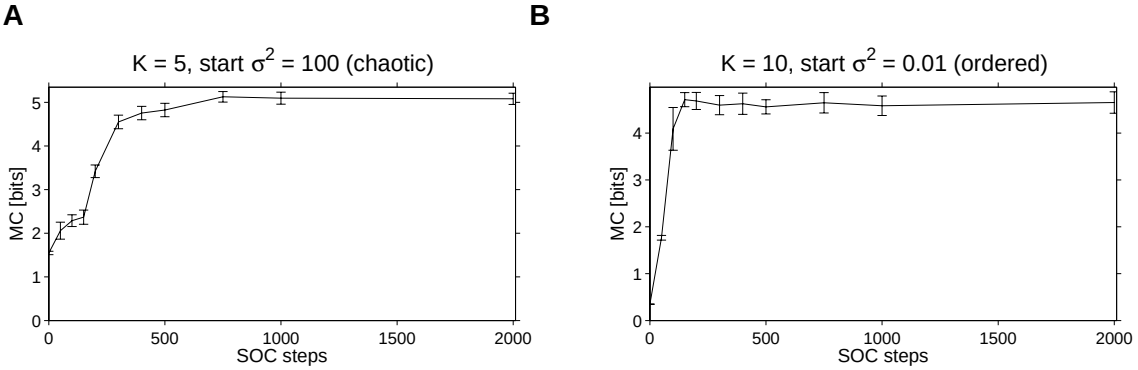


Figure 5: Time evolution of the performance with activated SOC rule (4). **A** The plot shows the memory capacity MC (see Fig. 3 on the 3-bit parity task averaged over 25 networks ($\pm$ standard deviation as errorbars) evaluated at the indicated time steps. At each evaluation time step the network weights were fixed and the MC was measured as in Fig. 3 by training the corresponding readouts from scratch. The networks were initialized in the chaotic regime. **B** Same as in A but for $K = 10$ and networks initialized in the ordered regime.

Since critical dynamics are better suited for infomation processing (see Fig. 3) it is expected that the performance on the 3-bit parity task improves due to SOC. This is confirmed in Fig. 5 which shows how the memory capacity MC (defined in Fig. 3) grows for networks that were initialized in the chaotic and ordered regime respectively.

7 Discussion

We developed a mean-field theory for input-driven networks which allows to determinate the position of the transition line between ordered and chaotic dynamics with respect to the parameters controlling the network connectivity and input statistics. Based on this theory we proposed a complexity measure (called NM-separation) which assumes its highest values at the critical line and shows a clear correlation with the computational power for real-time time-series processing. These results provide further evidence for the idea of "computation at the edge of chaos" [1] and support the hypothesis that dynamics near the critical line are a general property of input driven dynamical systems which support complex real-time computations.

Furthermore we have shown that a rule for synaptic scaling is able to adjust the weights of a network towards critical dynamics. In fact we have shown that networks adjusted by this rule show also enhanced computational capabilities. Hence combining task-specific optimization provided by (supervised) learning rules allows a system to gather task specific information while self-organizing its dynamics towards criticality in order to be able to react flexible to incoming signals.

References

[1] C. G. Langton. Computation at the edge of chaos. *Physica D*, 42, 1990.

[2] B. Derrida and Y. Pomeau. Random networks of automata: A simple annealed approximation. *Europhys. Lett.*, 1:45–52, 1986.

[3] B. Derrida. Dynamical phase transition in non-symmetric spin glasses. *J. Phys. A: Math. Gen.*, 20:721–725, 1987.

[4] N. Bertschinger and T. Natschläger. Real-time computation at the edge of chaos in recurrent neural networks. *Neural Computation*, 16(7):1413–1436, 2004.

[5] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11), 2002.

[6] H. Jaeger and H. Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.

[7] S. Bornholdt and T. Röhl. Self-organized critical neural networks. *Physical Review E*, 67:066118, 2003.

[8] N. Bertschinger and T. Natschläger. Details to the mean-field theory for randomly connected recurrent networks of threshold gates, 2004. electronically available via http://www.igi.tugraz.at/tnatschl/edge-of-chaos/mean-field-techreport.pdf.

Evaluation and interpretability-preserving optimization of a fuzzy ordered classifier

Ester Van Broekhoven[1], Veronique Adriaenssens[2] and Bernard De Baets[1]
[1] Department of Applied Mathematics, Biometrics and Process Control
Ghent University, Coupure links 653, B-9000 Gent, Belgium
[2] Department of Applied Ecology and Environmental Biology
Ghent University, Jozef Plateaustraat 22, B-9000 Gent, Belgium
E-mail: {Ester.VanBroekhoven,Veronique.Adriaenssens,Bernard.DeBaets}@UGent.be

***Abstract*— Fuzzy ordered classifiers were used to assign fuzzy labels to river sites expressing their suitability as a habitat for a certain macro-invertebrate taxon, given up to three abiotic properties of the considered river site. The models were built using expert knowledge and evaluated on data collected in the Province of Overijssel in the Netherlands. Apart from a performance measure for crisp classifiers common in the aquatic ecology domain, the percentage of correctly classified instances (% CCI), two performance measures for fuzzy (ordered) classifiers are introduced in this paper: the percentage of correctly fuzzy classified instances (% CFCI) and the average deviation (AD). Furthermore, results of an interpretability-preserving genetic optimization of the linguistic terms are presented.**

I. Introduction

Habitat suitability models describe which abiotic conditions are appropriate for a certain taxon or species to establish a population. In this study macro-invertebrates are considered, small animals living in aquatic ecosystems who can still be observed with the naked eye and whose abundances are a measure for biological water quality [1]. Habitat suitability models are meant to support decisions in river management, therefore Mamdani-Assilian models are appropriate model types as they allow for interpretability and uncertainty estimation by the end-user, as well as straightforward incorporation of new knowledge [2], [3].

As this modelling problem asked for a model that gives a shaded indication to what degree a certain river site is suitable as habitat for a certain macro-invertebrate taxon, we opted for fuzzy classifiers, instead of classical models with crisp outputs or crisp classifiers. A more detailed description of the habitat-suitability models, built using expert knowledge, is given in Section II.

In Section III the EKOO data set [4] on which the models were evaluated is discussed. The three measures used to evaluate the models, percentage of correctly classified instances (% CCI), percentage of correctly fuzzy classified instances (% CFCI) and average deviation (AD) are presented in Section IV. Section V deals with the different aspects of the optimization of the linguistic terms: the selection of the models to be optimized, the properties of the genetic algorithm and the obtained results. Finally, conclusions and further work are summarized in Section VI.

II. Habitat suitability models

For 86 macro-invertebrate taxa, habitat suitability models were built using expert knowledge, which comprises:

- the selection of variables,
- the assignment of linguistic values and corresponding membership functions to all variables and,
- the construction of rule bases.

The selected input variables should be of high ecological importance for the macro-invertebrate taxa under study as well as for the whole macro-invertebrate community and should be of importance for river management. Furthermore, knowledge about their influence on macro-invertebrates should be available and, of course, the variables should be included in the EKOO data set. With these criteria in mind, six variables were selected: two variables related to the dimension of the river, stream width and stream velocity, and four variables related to human impact (caused by agricultural activities), ammonium concentration, nitrate concentration, phosphate concentration and conductivity. For each macro-invertebrate taxon, four different models were constructed, an A-model, an N-model, a P-model and a C-model, containing stream width, stream velocity and either ammonium concentration (A), nitrate concentration (N), phosphate concentration (P) or conductivity (C) as input variables. The occurrence of some macro-invertebrate taxa is independent of the stream width. In these models stream width is not included and only two input variables are used. To all variables three to five linguistic values are assigned, defined by trapezoidal membership functions forming a Ruspini partition [5].

The abundance is used as output variable, as the EKOO data set contains the number of sampled individuals of the 86 taxa and it can be considered proportional to a sites suitability to establish a population. As neither a crisp abundance value, nor a crisp classification seemed in this case appropriate as model output, we opted for a fuzzy classification. Four linguistic values were assigned to the variable abundance: $absent$, low, $moderate$ and $high$. They are defined by the membership functions shown in Fig. 1. The model output $\mathbf{y}_{\text{model}}$ is a set of four values ranging between 0 and 1 and summing up to 1: $\{(absent,\ A_1(\mathbf{y}_{\text{model}}))$, $(low,\ A_2(\mathbf{y}_{\text{model}}))$, $(moderate,\ A_3(\mathbf{y}_{\text{model}}))$, $(high,\ A_4(\mathbf{y}_{\text{model}}))\}$.

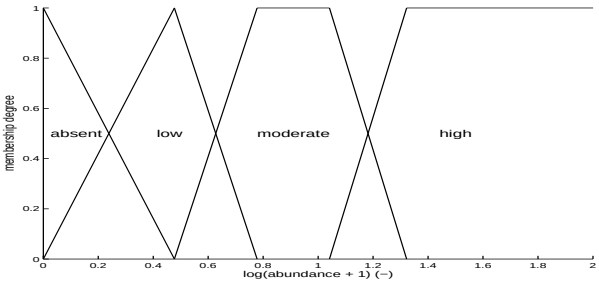


Fig. 1. Definition of the fuzzy abundance classes through membership functions

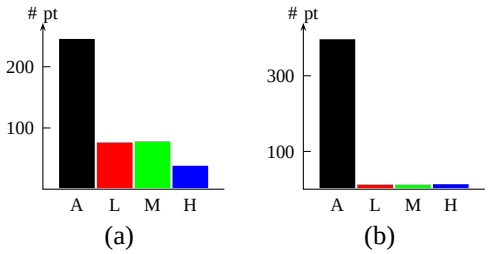


Fig. 2. Distribution of the data points among crisp abundances classes $absent$, low, $moderate$ and $high$ for (a) *Proasellus meridianus* and (b) *Plectronemia conspersa* (see Eq. (1) for the exact defuzzification procedure)

All constructed rule bases were complete and contained rules of the following type:

IF $width$ IS $upper\ course\ stream$
AND $velocity$ IS low
AND $nitrate\ concentration$ IS $eutrophic$
THEN $abundance$ IS $absent$

If two adjacent linguistic values of a variable yield the same model output for all combinations of linguistic values of the other input variables the corresponding rules are merged and a new linguistic value is introduced described by the convex hull of the membership functions of the original linguistic values. The minimum t-norm was applied for the conjunction. For each linguistic abundance value, the maximum fulfilment degree of the rules containing the linguistic abundance value in their consequent is determined. Finally, the model output is obtained by normalizing these maximum fulfilment degrees.

III. EKOO DATA SET

The EKOO data set contains values for 445 sites in running waters in the Province of Overijssel in the Netherlands for abiotic parameters, as river width, dissolved oxygen concentration, pH, ammonium concentration, as well as the number of sampled individuals of several macro-invertebrate taxa. Hours of field work and meticulous determination in the lab of the sampled animals were needed to obtain this data set, which makes it a large data set in its domain, but unfortunately still rather small for model evaluation and certainly for model identification purposes. Apart from being sparse, the data hold another awkward property typical to their origin: they are characterized by high variability due to for instance seasonal variations, weather differences at sampling moment and different sediments. This results in data holding similar river conditions but completely different registered abundances, i.e. in ambiguous data. Furthermore, for all 86 taxa considered in this study at a vast majority of the 445 sites no individuals were recorded, as illustrated for *Proasellus meridianus* and *Plectronemia conspersa* in Fig. 2.

IV. EVALUATION OF FUZZY ORDERED CLASSIFIERS

A. Format of the reference output

In order to compare the output obtained with the fuzzy ordered classifiers with the information in the EKOO data set, model and reference output should have the same format. In this study the membership degrees of the crisp abundances values in the data set to the linguistic abundance values are used as reference output.

B. Three performance measures

In this section three performance measures applied in this study are introduced. In the formulae below, N is the number of data points, n the number of fuzzy classes, $A_i(\mathbf{y}_{\text{data},j})$ the membership degree of the j^{th} output to the i^{th} linguistic output value and $A_i(\mathbf{y}_{\text{model},j})$ the membership degree to i^{th} linguistic output value obtained as model output for the j^{th} input of the data set.

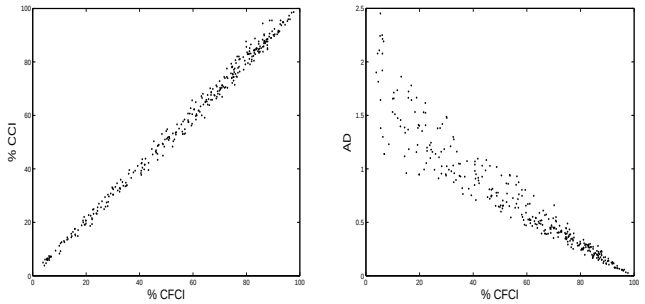
1) Percentage of correctly classified instances: In aquatic ecology, the percentage of correctly classified instances (% CCI), is used as a standard to compare the performance of crisp classifiers. Correctly classified data points have a contribution of 1 to the global performance, while data points assigned to a wrong class have a contribution of 0. In order to be able to compare our fuzzy classifiers with crisp classifiers in literature, the outputs were defuzzified and the % CCI was calculated as follows:

$$\%\ \text{CCI} = \frac{100}{N}\sum_{j=1}^{N}\left(1 - \frac{1}{2}\sum_{i=1}^{n}\left|A_{crisp,i}(\mathbf{y}_{\text{data},j}) - A_{crisp,i}(\mathbf{y}_{\text{model},j})\right|\right) \quad (1)$$

with

$$A_{crisp,i}(\mathbf{y}) = \begin{cases} 1 & \text{if } i = \min\{k | A_k(\mathbf{y}) = \max_{l=1}^{n} A_l(\mathbf{y})\}, \\ 0 & \text{otherwise.} \end{cases}$$

2) Percentage of correctly fuzzy classified instances: As we were dealing with fuzzy classifiers, we defined a new performance measure inspired by the % CCI and similar to the measure presented in [6]: the percentage of correctly fuzzy classified instances (% CFCI). If the model output is identical



(a) (b)

Fig. 3. Comparison of the % CFCI-values to (a) the % CCI- and (b) AD-values

to the reference output, the data point has a contribution of 1 to the global performance. As long as there are classes to which both model output and reference output have a non-zero membership degree, the corresponding data point has a positive contribution. Only if there exists no class to which both model output and reference output have a non-zero membership degree, the corresponding data point has a contribution of 0 to the global performance.

$$\%\ \text{CFCI} = \frac{100}{N}\sum_{j=1}^{N}\left(1-\frac{1}{2}\sum_{i=1}^{n}\left|A_i(\mathbf{y}_{\text{data},j})-A_i(\mathbf{y}_{\text{model},j})\right|\right).$$

3) Average deviation: The % CFCI has the advantage that it can be understood intuitively. However, it is not an appropriate objective function for the optimization of an ordered fuzzy classifier, as % CFCI is not sensitive for the position of the classes were the wrong classification occurs. Therefore another performance measure for fuzzy classifiers with an ordered set of classes is introduced, returning the average deviation (AD) between the position of the class obtained with the model and the position of the class stored in the reference data set. The AD varies from 0 to $n-1$ and is calculated as follows:

$$\text{AD} = \frac{1}{N}\sum_{j=1}^{N}\sum_{i=1}^{n-1}\left|\sum_{k=1}^{i}A_k(\mathbf{y}_{\text{data},j})-\sum_{k=1}^{i}A_k(\mathbf{y}_{\text{model},j})\right|.$$

C. Model performance

In Fig. 3 the three performance values obtained for the four models of the 86 macro-invertebrate taxa are plotted. One sees that similar values are obtained for % CCI as for its fuzzy alternative, % CFCI, and that AD tends to decrease with increasing % CFCI. The % CFCI of the A-, N-, P- and C-models of all taxa are shown in Fig. 4. For almost all taxa, higher % CFCI-values are obtained for models including nitrate or phosphate concentration as input variable than for those including ammonium concentration or conductivity.

V. Optimization of the linguistic terms

A. Model selection

As mentioned in Section III, the EKOO data set is characterized by ambiguous data as well as by a highly non-uniform distribution of the data among the four abundance classes $absent$, low, $moderate$ and $high$. As the more different phenomena described by the model are included in a data set, the more appropriate the data set is for optimization, data sets with the most uniform distribution among the crisp abundance classes were selected for optimization. As a measure for the uniformity of the distribution, entropy was used:

$$entropy = -\frac{1}{\log_2 n}\sum_{i=1}^{n} p_i \cdot \log_2 p_i$$

with

$$p_i = \frac{1}{N}\sum_{j=1}^{N} A_{crisp,i}(\mathbf{y}_{\text{data},j})$$

$$p_i \cdot \log_2 p_i = 0, \text{ if } p_i = 0$$

The entropy is 1 for a uniform distribution and 0 if all data points are assigned to the same abundance class. For *Proasellus meridianus* and *Plectronemia conspersa*, of which the data point distributions are shown in Fig. 2, respectively an entropy of 0.834 and 0.322 is obtained. In Fig. 5 the entropy of the data distribution among the abundance classes for the 86 macro-invertebrate taxa is plotted as a function of the % CFCI of the A-model of the corresponding taxon. The figure gives an insight into the obtained values for the performance measures. One can see that a *good* performance according to the values of the performance measure often coincides with a low entropy. These *good* performing models are all models of macro-invertebrate taxa of which no individuals were collected at almost all 445 sampled sites and which are therefore not really evaluated by the data set. The twelve models selected for optimization are indicated with a box.

B. Properties of the genetic algorithm

During the optimization, the membership functions of the input variables were searched with a genetic algorithm [7]. Two optimizations were carried out: a bounded and a free optimization. During the bounded optimization the kernels of the optimized membership functions are always subsets of the 0.5-cuts of the corresponding original membership functions (as illustrated in Fig. 6), whereas during the free optimization only the number of membership functions is fixed for each input variable.

During the search, each obtained model was evaluated on each of the 445 data points, using a weighted average deviation (wAD) in which the weights guarantee that each region of the input space defined by the 0.5-cuts of the membership functions defined by the expert has the same contribution to the fitness:

$$\text{wAD} = \sum_{j=1}^{N} w_j \cdot \sum_{i=1}^{n-1}\left|\sum_{k=1}^{i}A_k(\mathbf{y}_{\text{data},j})-\sum_{k=1}^{i}A_k(\mathbf{y}_{\text{model},j})\right|$$

with

$$w_j = \frac{1}{N_j \cdot n_{regions}}.$$

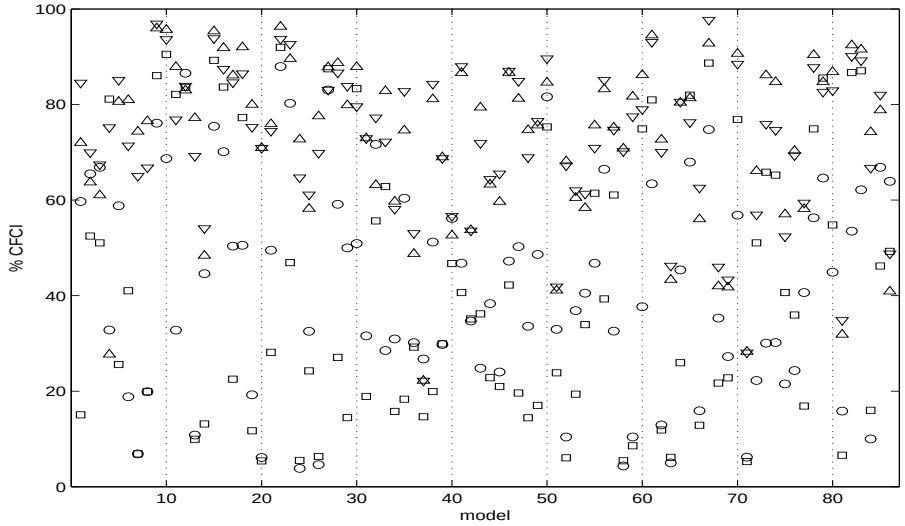


Fig. 4. Percentage correctly fuzzy classified instances for the A-model (□), N-model (▽), P-model (△) and C-model (◯) for the 86 macro-invertebrates

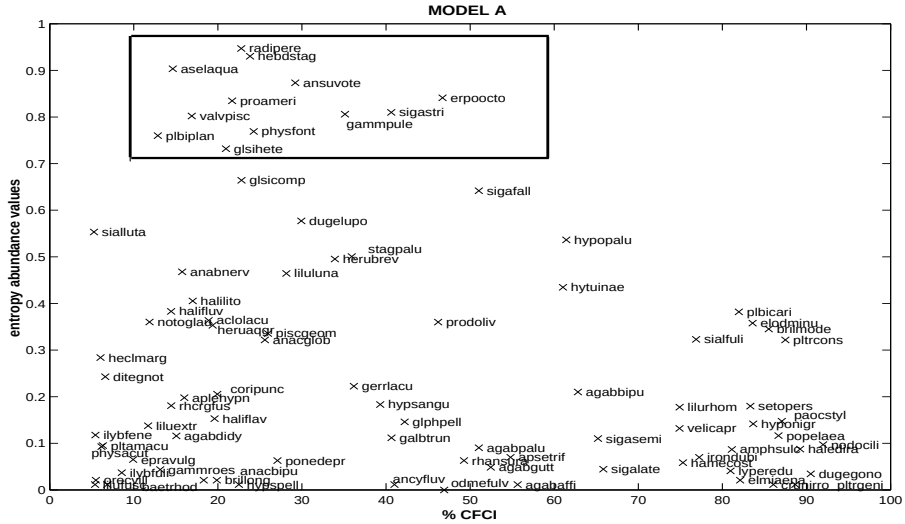


Fig. 5. Entropy and % CFCI of the 86 models including the ammonium concentration as in input variable. The 12 models selected for optimization are indicated with a box.

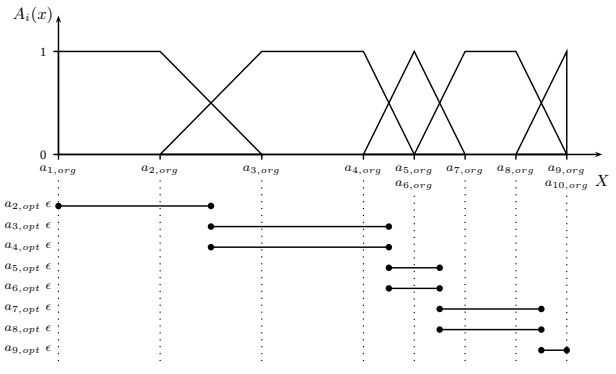


Fig. 6. Illustration of the boundaries used for the membership functions parameters during the bounded simulation

In the definition of the weights w_j, N_j is the number of data points in the same region of the input space as the j^{th} input of the data set and $n_{regions}$ is the number of regions in which the input space is divided.

The membership function parameters were coded as binary strings of respectively 7 and 10 bits per parameter for respectively the bounded and free optimization. A population contained 100 individuals and each generation parents were selected by tournament selection and recombined using uniform crossover (crossover probability = 0.95) and mutation (mutation probability = (length binary string)$^{-1}$). Furthermore, elitism was applied in the algorithm. The genetic algorithm was stopped if only small improvements of the fitness of the best individual ($\triangle$ fitness < 0.001) were obtained during the last 50 consecutive generations or if the 1000^{th} generation

was reached. Hundred repetitions were carried out for each optimization and the model with the highest % CFCI among the 100 candidate models was retained as result of the optimization.

C. Optimization results

The results obtained for the twelve selected models are summarized in Fig. 7. One expects the model obtained through free optimization to have a higher % CFCI than the corresponding model obtained through bounded optimization, which on its turn is expected to score better than the original model. This is indeed the case for all optimized A- and C-models and all P-models, except for *Physa fontinalis*. However, for *Anisus vortex*, *Erpobdella octoculata*, *Gammarus pulex*, *Glossiphonia heteroclita*, *Physa fontinalis* and *Radix peregra* the order of the % CFCI-values of the original, bounded and freely optimized N-models does not agree the expected order. Three reasons can be given for these anomalous results. First of all, the values taken by the membership function parameters in the optimized models are restricted to a limited set of values due to the use of binary coding. Furthermore, the ambiguity of the data can be blamed and finally, the fact that the original models including nitrate concentration scored best among the four original models of these taxa, could indicate lack of room for further optimization of these models.

In Figs. 8 and 9 the results obtained for the A-model of *Proasellus meridianus* are shown. Note that the membership function describing the oligosaprobic (including α,β-oligosaprobic) conditions in the original model has such a small support that it can hardly be noticed. In Fig. 8 one sees that the membership functions of the velocity value low and the $oligosaprobic$ conditions are extended towards higher velocities and ammonium concentrations respectively. The membership functions in Fig. 8(c) no longer reflect the meaning given by the experts to the linguistic values. During the bounded optimization the extension is however limited by the constraints described in Section V-B. As shown in Fig. 9, by extension of the support of these linguistic values, more data points and in particular more data points belonging to the abundance class Absent, fire the rule

IF vel IS low AND $ammon$ IS $oligotrophic$
THEN $abundance$ IS $absent$,

instead of the rules

IF vel IS low AND $ammon$ IS $\beta - mesotrophic$
THEN $abundance$ IS low ,

IF vel IS $moderate$ AND $ammon$ IS $oligotrophic$
THEN $abundance$ IS low ,

IF vel IS $moderate$ AND $ammon$ IS $\beta - mesotrophic$
THEN $abundance$ IS $moderate$,

which results in a better score for the used fitness wAD as well as for the other performance measures % CCI, % CFCI and AD.

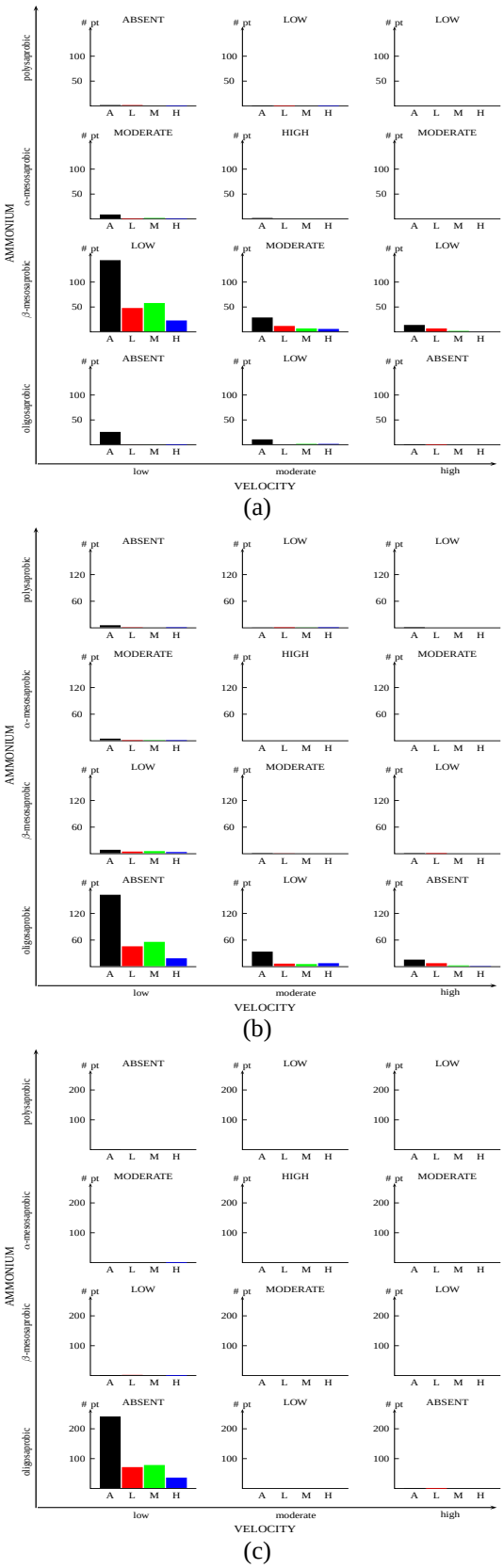


Fig. 9. Distribution of the data points among the different subregions of the input space defined by 0.5-cuts of the membership functions of (a) the original model, (b) the model obtained through bounded optimization and (c) free optimization of the A-model of *Proasellus meridianus*

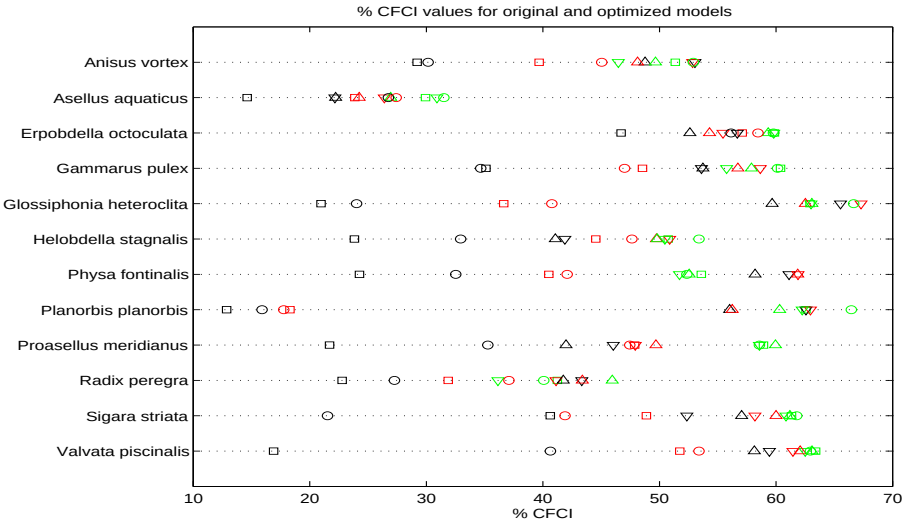
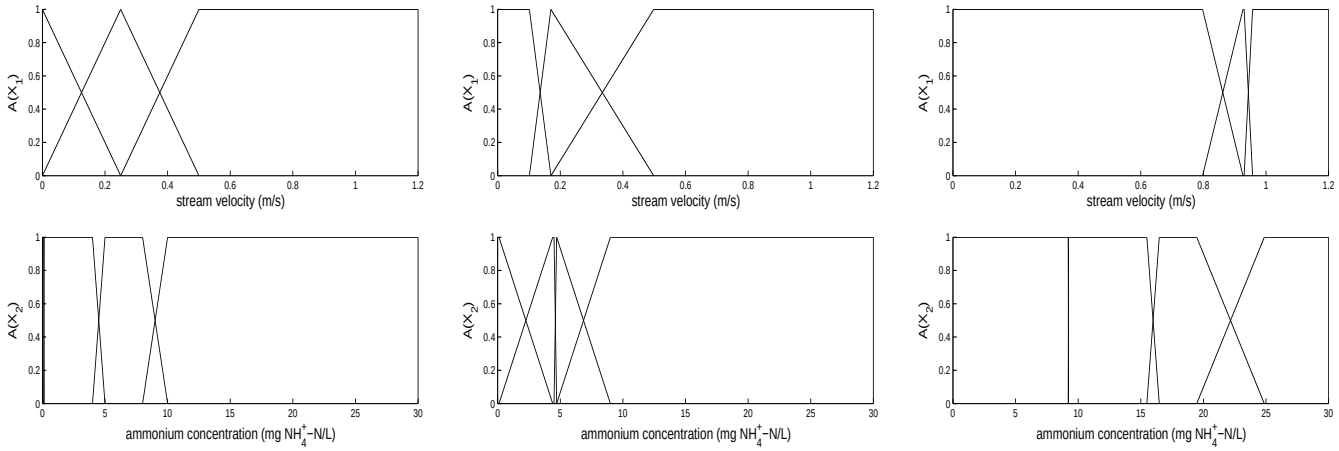


Fig. 7. Percentage correctly fuzzy classified instances for the original, bounded and free optimized models for the twelve selected taxa. Legend: □: A-model, ▽: N-model, △: P-model and ◯: C-model; black: original model, red: bounded optimization and green: free optimization



(a) (b) (c)

Fig. 8. Membership functions of (a) the original model, (b) the model obtained through bounded optimization and (c) free optimization of the A-model of *Proasellus meridianus*

VI. Conclusions

In this study fuzzy ordered classifiers were used to classify river sites according to their suitability as a habitat for macro-invertebrates. The classifiers were evaluated using data collected in the Province of Overijssel in the Netherlands. Two performance measures were introduced in this paper: the percentage of correctly fuzzy classified instances, % CFCI, for ordered and not-ordered fuzzy classification, and the average deviation, AD, for ordered fuzzy classification. Due to its ambiguity, the available data set was not found appropriate for a purely data-driven model identification, expert knowledge was a prerequisite to build interpretable models. One type of interpretability-preserving data-driven optimization was introduced and applied using a genetic algorithm.

References

[1] N. De Pauw and G. Vanhooren, "Method for biological quality assessment of water courses in belgium," *Hydrobiologia*, vol. 100, pp. 153–168, 1983.

[2] J. Casillas, O. Cordón, F. Herrera, and L. Magdalena, *Interpretability Issues in Fuzzy Modeling*, ser. Studies in Fuzziness and Soft Computing. Heidelberg: Springer Verlag, 2003, vol. 128.

[3] ——, *Accuracy Improvements in Fuzzy Modeling*, ser. Studies in Fuzziness and Soft Computing. Heidelberg: Springer Verlag, 2003, vol. 129.

[4] P. Verdonschot, "Ecological characterization of surface waters in the province of overijssel (The Netherlands)," Ph.D. dissertation, Landbouwuniversiteit Wageningen, Wageningen, The Netherlands, 1990.

[5] E. Ruspini, "A new approach to clustering," *Information and Control*, vol. 15, pp. 22–32, 1969.

[6] U. Bodenhofer and E. Klement, "Genetic optimization of fuzzy classification systems — A case study," in *Computational Intelligence in Theory and Practice*, ser. Advances in Soft Computing, B. Reusch and K.-H. Temme, Eds. Heidelberg: Physica Verlag, 2001, pp. 183–200.

[7] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman, 1989.

FLEXFIS: A Variant for Incremental Learning of Takagi-Sugeno Fuzzy Systems

Edwin Lughofer, Erich Peter Klement
{edwin.lughofer,ep.klement}@jku.at
Johannes Kepler University Linz
Altenbergerstrasse 69, A-4040 Linz, Austria

***Abstract*— In this paper a new algorithm for the incremental learning of specific data-driven models, namely so-called Takagi-Sugeno fuzzy systems, is introduced. The new open-loop learning approach includes not only adaptation of linear parameters in fuzzy systems appearing in the rule consequents, but also sample mode adaptation of premise parameters appearing in the membership functions (i.e. fuzzy sets) together with a rule learning strategy. In this sense the proposed method is applicable for fast model training tasks in various industrial processes, whenever there is a demand of online system identification in order to apply models representing nonlinear system behaviors to system monitoring, online fault detection or open-loop control. An evaluation of the incremental learning algorithm is included at the end of the paper, where a comparison between conventional closed-loop modelling methods for fuzzy systems and the incremental learning method (also called adaptation in open-loop) demonstrated in this paper is made with respect to model qualities and computation time. This evaluation will be based on high dimensional data coming from an industrial measuring process as well as from a known source in the internet, which should underline the usage of the new method for fast online identification tasks.**

***Index Terms*— Incremental learning, sample mode adaptation, Takagi-Sugeno fuzzy systems, rule learning, model qualities, online identification**

I. Introduction

Nowadays Takagi-Sugeno fuzzy systems play an important role for system modelling and identification tasks, as they possess the behavior to be capable for approximating any nonlinear dependency to a certain degree of accuracy between some physical, chemical or medical variables occurring in the real world. They are also applied in applications areas such as system analysis (as they gain linguistic interpretable models in form of rule bases and therefore may yield a better understanding of some underlying system behaviors), prediction, control (as they provide a good interpretation about local behaviors of a control system through observing the steepness of the hyper-planes in different directions), fault detection or simply simulation. A specific task is the so-called online identification of Takagi-Sugeno fuzzy systems, where this type of fuzzy systems should be trained in an incremental manner, i.e. stepwise with newly loaded data points without taking into account the former loaded ones. Opposed to a re-building of the systems from time to time by sending all the data measured so far into a conventional closed-loop training algorithm, the incremental learning approach guarantees a fast training of the fuzzy systems, which can be indispensable within an online system. Other requirements to incremental learning includes refinement of already existing knowledge-based models with data (the achieved models are then also called *grey box models*), preventing a virtual memory overload in the case of a very huge amount of data (e.g. a database containing four million measurements and 100 different continuous variables) and improving process security by preventing extrapolation to new operating conditions and by filling large interpolation holes. The last point is essential when a re-building from time to time is impossible due to computational performance reasons and an initially built fuzzy system does not cover the complete region of the input space sufficiently.

As Takagi-Sugeno fuzzy systems and a special case of them, the so-called Sugeno fuzzy systems, possess linear parameters in the rule consequent functions (see Section II for a detailed definition), a recursive algorithm as it is also applied for ARX and ARMAX regression models [1] is quite often applied in literature [2], [3], [4] and will be also described shortly in Section III as it acts as an important part within the complete incremental learning approach of fuzzy systems demonstrated throughout this paper. However, as it will be also demonstrated in Section III, the adaptation of the linear rule consequent parameters alone is not satisfactory, as it is not capable to incorporate a description about new system behaviors or new operating conditions by means of additional fuzzy sets and rules in the input space. A possibility to overcome this drawback is demonstrated in [5], where a batch mode adaptation strategy within a sliding window including the last few 100 data points is carried out. An extension of this approach to sample mode adaptation is stated in [6]. An alternative to this approach is presented in [7], Section 14.6.2, by applying the closed-loop training method *LOLIMOT = LOcal LInear MOdel Tree* [8], which ensures nicely interpretable fuzzy partitions. However, this approach lacks of being capable for incorporating new operating conditions represented by new incoming data appearing outside the prior estimated range. In [4] it is described that *lazy learning* [9] is a good alternative to the adaptation of nonlinear parameters in a fuzzy system. However, its outcome is a kind of local approximator as it only uses the nearest local points for building up a model, so no global model yielding an interpretation of the entire process behavior is achieved.

In this paper a new approach for the incremental learning of Takagi-Sugeno fuzzy systems is demonstrated, which exploits

open-loop capable clustering methods for online training of fuzzy partitions as well as rule bases, with the possibility of adjoining new fuzzy sets and rules whenever there is a demand for it due to the nature of the newly loaded data. This online training strategy for the nonlinear part is combined with the recursive adaptation of linear parameters appearing in the rule consequents to a complete data-driven incremental learning algorithm. In doing so, it is practicable for taking into account each new incoming data point separately for the model building process and therefore feasible for online learning and identification tasks of various industrial processes, where models should be up-to-date as fast as possible.

II. Problem Statement

A. Definition of a Takagi-Sugeno Fuzzy System

The product t-norm together with Gaussian functions leads to some favorable properties, namely steady differentiable models, equivalency to radial basis functions neural networks [10], [11], favorable interpolation properties due to infinite support and an easy extraction of the parameters for the Gaussian fuzzy sets (which is for instance not the case for sigmoidal functions, which also possess infinite support). Due to these reasons, this combination is taken into account, leading to the specific case of the Takagi-Sugeno fuzzy system with multiple input variables $(x_1, ..., x_p)$ and a single output variable y defined by

$$\hat{f}(\vec{x}) = \hat{y} = \sum_{i=1}^{C} l_i \Psi_i(\vec{x}) \tag{1}$$

with the basis functions

$$\Psi_i(\vec{x}) = \frac{e^{-\frac{1}{2}\sum_{j=1}^{p}\frac{(x_j - c_{ij})^2}{\sigma_{ij}^2}}}{\sum_{k=1}^{C} e^{-\frac{1}{2}\sum_{j=1}^{p}\frac{(x_j - c_{kj})^2}{\sigma_{kj}^2}}} \tag{2}$$

and consequent functions

$$l_i = w_{i0} + w_{i1}x_1 + w_{i2}x_2 + ... + w_{ip}x_p \tag{3}$$

This special form of a Takagi-Sugeno fuzzy system is also called *FBFN (i.e. Fuzzy Basis Function Networks)*.

B. What to adapt?

When inspecting the formulation of a Takagi-Sugeno fuzzy system as in (1) we have to realize that principally three different kinds of components may be adapted in open loop manner:

- Consequent Parameters: they appear in the rules consequents as output weights $(w_{i0}, w_{i1}, ..., w_{ip})$
- Premise Parameters: they appear in the input membership functions as centers (c_{ij}) and widths (σ_{ij})
- Rule Base: this concerns not only the amount of rules given by the summation limit C in (1), but also the number of fuzzy sets per input dimension, which influences the amount of rules and readability of the whole system.

Consequent parameters are linear parameters, premise parameters are nonlinear ones, hence two completely different approaches for adaptation are described in the following sections within this section. Within all approaches the adjustment of the parameters should be carried out in a way such that previously learned relationships should not be forgotten while approximating new relationships (described by the new incoming points) as narrow as possible, meaning that the approximation function obtained by an incremental learning approach should not cause a significantly worse quality than an approximation obtained from closed-loop model building.

III. Adaptation of Rules' Consequents

A. Global Estimation and Adaptation of Rules' Consequents

The goal of global least squares estimation is, to minimize the error between the measured values y_k and the estimated values $\hat{y}_k$ of the output variable, where $k = 1, ..., N$ and N the number of available data points. This leads us to the following minimization problem when applying the quadratic distance error measure to the Takagi-Sugeno fuzzy system as formulated in (1):

$$J = \sum_{k=1}^{N}(y(k) - \sum_{i=1}^{C} l_i \Psi_i(\vec{x}(k)))^2 = \min_{\vec{w}}! \tag{4}$$

which can be uniquely solved by derivation the objective function after all linear parameters and solving the resulting linear equation system, when taking into account, that $\hat{y}$ can be rewritten as $R\vec{w}$, where $\vec{w}$ contains all the linear parameters for all rules to be estimated and the matrix R denotes the regression matrix containing the regressors

$$\vec{r}_i(k) = [\Psi_i(\vec{x}(k))x_1(k)\Psi_i(\vec{x}(k)) \dots x_p(k)\Psi_i(\vec{x}(k))]$$

for all C rules and $k = 1, \dots, N$ data points, where $x_i(k)$ is the ith column of the row vector $\vec{x}$ in point k. Setting the derivative with respect to the linear parameter vector $\vec{w}$ to 0, the parameters can be obtained by

$$\hat{\vec{w}} = (R^T R)^{-1} R^T \vec{y} \tag{5}$$

The great drawback of this method is that all data samples $(\vec{x}(1), y(1)), (\vec{x}(2), y(2)), ..., (\vec{x}(N), y(N))$ have to be sent into the algorithm at once, otherwise it is not working. A recursive formulation of the LS method for estimating linear consequent parameters, the so-called *recursive least squares (RLS)* [1], [12] overcomes this drawback as it calculates a new update for the linear parameters $\vec{w}$ each time a new data point comes in. The basic idea of the *RLS* algorithm is to compute the new parameter estimate $\hat{\vec{w}}(k+1)$ at time instant $k+1$ by adding some correction vector to the previous parameter estimate $\hat{\vec{w}}(k)$ at time instant k, leading finally to the following recursive estimator for the linear rule consequent parameters:

$$\hat{\vec{w}}(k+1) = \hat{\vec{w}}(k) + P(k+1)\vec{r}(k+1)(y(k+1) - \vec{r}^T(k+1)\hat{\vec{w}}(k)) \tag{6}$$

with $P(k+1)$ the inverse Hesse matrix $(R^T R)^{-1}$ at time instance $k+1$. Note, that the amount of correction is proportional to the prediction error.

The RLS algorithm requires the inversion of the Hesse Matrix H or P, respectively, hence the complexity is still $O(M^3)$, with M being the number of degrees of freedom or the number of parameters, respectively, so $M = C(p+1)$ with

C the number of rules and p the input dimension. Utilizing a specific matrix-inversion theorem, the inversion of the Hesse Matrix can be avoided and the computational complexity reduced to $O(M^2)$. The following formulas are obtained for RLS:

$$\hat{\vec{w}}(k+1) = \hat{\vec{w}}(k) + \gamma(k)(y(k+1) - \vec{r}^T(k+1)\hat{\vec{w}}(k)) \quad (7)$$

with the correction vector

$$\gamma(k) = P(k+1)\vec{r}(k+1) = \frac{P(k)\vec{r}(k+1)}{1 + \vec{r}^T(k+1)P(k)\vec{r}(k+1)} \quad (8)$$

$P(k+1)$ can be computed recursively by:

$$P(k+1) = (I - \gamma(k)\vec{r}^T(k+1))P(k) \quad (9)$$

B. Local Estimation and Adaptation of Rules' Consequents

Opposed to the global estimation, for the local approach M separate local estimations and consequently adaptations are carried out for the $p+1$ parameters of each local linear model, hence for each rule. A local linear model with the output $\hat{\vec{y}}_i = [\hat{y}_i(1) \;\; \hat{y}_i(2) \;\; \ldots \;\; \hat{y}_i(N)]^T$

$$\hat{\vec{y}}_i \quad = \quad R_i\vec{w}_i \quad (10)$$

where $\vec{w}_i$ the linear parameter vector and R_i the regression matrix for the ith rule, which just contains all regressor values of the original data points and is therefore the same for all rules, is valid only in the region where the associated *basis function* $\Psi_i(.)$ is close to 1, which will be the case close to the center of $\Psi_i(.)$ Consequently, it is straightforward to apply a weighted least squares optimization where the weighting factors are denoted by the *basis function* values, i.e.

$$J_i = \sum_{k=1}^{N} \Psi_i(\vec{x}(k))e_i^2(k) \longrightarrow \min_{w_i} \quad (11)$$

where $e_i(k) = y(k) - \hat{y}_i(k)$ represent the local linear model error in the kth point, whose solution leads to the so-called *weighted least squares* approach

$$\hat{\vec{w}}_i = (R_i^T Q_i R_i)^{-1} R_i^T Q_i \vec{y} \quad (12)$$

with Q_i the weighting matrix for the ith rule containing its basis function values at each data point.

The deduction of the open-loop version of the *weighted least squares*, the so-called *recursive weighted least squares = RWLS* is the same as in the previous section for the least squares approach, leading to the following update formulas for the linear consequent parameters of the ith rule:

$$\hat{\vec{w}}_i(k+1) = \hat{\vec{w}}_i(k) + \gamma(k)(y(k+1) - \vec{r}^T(k+1)\hat{\vec{w}}_i(k)) \quad (13)$$

$$\gamma(k) = \frac{P_i(k)\vec{r}(k+1)}{\frac{1}{\Psi_i(\vec{x}(k+1))} + \vec{r}^T(k+1)P_i(k)\vec{r}(k+1)} \quad (14)$$

$$P_i(k+1) = (I - \gamma(k)\vec{r}^T(k+1))P_i(k) \quad (15)$$

with $P_i(k) = (R_i(k)^T Q_i(k) R_i(k))^{-1}$ the inverse weighted inverse Hesse matrix and $\vec{r}(k+1) = [1 \;\; x_1(k+1) \;\; x_2(k+1) \;\; \ldots \;\; x_p(k+1)]^T$ the regressor values of the $k+1$th data point, which is the same for all i rules.

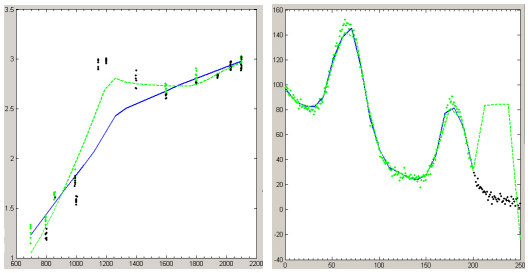


Fig. 1. Incorrect adapted model (dashed light line) due to new incoming data points (dark dots)

Based on a detailed comparison between the global and local approach it turned out that local approach is superior to global one in many aspects such as numerical stability (as dealing with inversion of smaller matrices), computational performance, the bias error when approximating from data with medium and high noise levels and transparency of the consequent functions (hyper-planes). Adaptation of the linear consequents alone is feasible whenever small holes in the original data set appear which are closed with new data. However, in practical situations there can be two main reasons, that this strategy fails, namely if data holes at the initial generation are too wide in order to guarantee enough flexibility of the fuzzy model later on or if operating conditions trigger data points in until then unexplored regions in the input space This fact is underlined when inspecting Figure 1 where the adaptation process, namely to extend the already generated models (solid lines) in the direction where the new incoming data points (dark dots) lie, fails, which is demonstrated by the dashed lines in both images.

IV. Adaptation of Premise Parameters and Rule Learning

Vector quantization is exploited in an open-loop variant and combined with the idea of ART-networks, i.e. to take into account the distance of newly loaded points to all the cluster centers generated so far. This distance is compared with a so-called vigilance parameter and if it is greater a new cluster is set, if this is not the case, the nearest cluster (also called winner neuron c_{win}) is updated for a new incoming point with respect to its width $\sigma_{win,j}$ by exploiting recursive variance formula [13] for each dimension j:

$$k_i\sigma_{win,j}^2 = (k_i - 1)\sigma_{win,j}^2 + k_i\Delta c_{win,j}^2 + (c_{win,j} - x_{kj})^2 \quad (16)$$

where $\Delta c_{win,j}$ is the distance of the old prototype to the new prototype of the winner neuron in the jth dimension and k_{win} is the amount of data points lying nearest to cluster c_{win} and can therefore be simply updated through counting. The center of the winner neuron is updated by

$$\vec{c}_{win}^{(new)} = \vec{c}_{win}^{(old)} + \eta(\vec{x} - \vec{c}_{win}^{(old)}) \quad (17)$$

with η the learning gain. A favorable choice of η with respect to a kind of convergence of the incremental learning algorithm over data samples (see next section) is given by

$$\eta_{win} = \frac{0.5}{k_{win}} \quad (18)$$

From the clusters the fuzzy sets are extracted by projection and the rules are obtained by simply connecting fuzzy sets belonging to one cluster with a t-norm. Hence, in the case a new cluster is born, automatically a new rule (the $C+1$th) is born:

$$\begin{aligned}\text{Rule}_{C+1}\ &:\text{IF } x_1 \text{ IS } \mu_{(C+1)1} \text{ AND } x_2 \text{ IS } \mu_{(C+2)j} \text{ AND...}\\ &\text{AND } x_p \text{ IS } \mu_{(C+1)p} \text{ THEN}\\ &w_{(C+1)0} + w_{(C+1)1}x_1 + ... + w_{(C+1)p}x_p\end{aligned}$$

with $\mu_{(C+1)j}$ described by $c_{(C+1)j}$ and $\sigma_{(C+1)j}$ set to:

$$\begin{aligned}c_{(C+1)j} &= x_{kj} \ \ \forall j \in \{1,...,p\}\\ \sigma_{(C+1)j} &= \epsilon * range(\vec{x_j}) \ \ \forall j \in \{1,...,p\}\end{aligned} \tag{19}$$

and where the linear weights $w_{(C+1)j}$ are set initially to 0 for start values to *RWLS*. From this point of view, a cluster update always belongs to a rule update. For details about this learning strategy of the antecedent part in a Takagi-Sugeno fuzzy system and evaluation results with respect to a performance comparison on 2-dimensional data with *conventional genfis2* (as implemented in MATLAB [14], [15]) refer to [16].

V. Inremental Learning of Takagi-Sugeno Fuzzy Models

The connection of the premise parameter and rule structure innovation algorithm as demonstrated in IV with the adaptation approaches for linear consequent parameters as described in III is carried out in sample mode manner. Hereby, we have to consider about three important points, namely the choice of the adaptation approach for the rule consequents, the eventuality of adaptation from scratch and the way how to connect. For the adaptation of rules consequents we choose the local approach, as it possesses a lot of advantages among the global one, see Section III. Regarding to adaptation from scratch we have to realize that there is one hitch when trying to build up the rule structures and fuzzy sets from scratch: usually in the case of a complete automatic data-driven generation of fuzzy models, the ranges of the input variables are not known or available. This makes the learning approaches for premise parameters and rules not applicable as a data normalization has to be carried out with the exploitation of the ranges of the input variables. In this sense, a significant amount of data points is important in order to be able to achieve a good first estimate of the ranges. From first glance, it looks straightforward to connect the two different approaches, hence either first to adapt the premise parameters and then second to adapt the consequent parameters of the rule which corresponds to the winner neuron or vice versa. However, when performing this straightforward connection, an instability of the linear consequent adaptation part may arise. This is due to the fact that a modification in the clusters causes a change in the projected fuzzy sets, which has an influence onto the already estimated consequent parameters as well as inverse Hesse matrix, such that correction terms should be incorporated before performing the local adaptation for the next ($k+1$th) data point. These correction terms will be denoted as $\vec{\delta}(k+1)$ for the linear parameters and $\Delta(k+1)$ for the inverse Hesse matrix. In the case of setting a new cluster and hence a new rule triggering new fuzzy sets by projection, the already estimated parameters as well as the inverse Hesse matrices are not disturbed as for all other rules the *weighted recursive least squares* is carried out separately. With these considerations we obtain the following incremental learning algorithm for Takagi-Sugeno fuzzy systems:

Algorithm 1: **FLEXFIS: Incremental Learning of Takagi-Sugeno fuzzy systems**

1) Collect k sufficient data points in order to yield a correct and stable approximation.
2) From these collected k data points generate an initial fuzzy model with *genfis2 extended*, hence with VQ-ART instead of subtractive clustering and local estimation approach for rules' consequents instead of global one
3) Take the next incoming data point (online case) or fetch out a data sample from a data matrix randomly or ordered (offline case): $\vec{x}_{k+1}$
4) Normalize cluster centers and widths as well as the current data point due to the ranges from the previous cycle This has the effect that new incoming data points lying significantly outside the already estimated range cause certainly a new rule.
5) Calculate the distance of the selected data point to all cluster centers by using a predefined distance measure. Here, Euclidian is used.
6) Elicit the cluster center which is closest to the data point by taking the minimum over all calculated distances $\rightarrow$ winner neuron c_{win}
7) **If** $\|\vec{x}_{k+1} - c_{win}\|_A \geq \rho$ with A the Euclidian norm
   a) Increase the number of clusters (i.e. rules) C, hence $C = C + 1$
   b) Start a new cluster at the $k+1$th point, hence $\vec{c}_C = \vec{x}_{k+1}$.
   c) Set the width of the new cluster to zero, hence $\vec{\sigma}_C = \vec{0}$
   d) Transfer cluster centers and widths back to original range with respect to each dimension j, hence
   $$\begin{aligned}c_{ij} &= c_{ij}(\max(\vec{x}_j) - \min(\vec{x}_j)) + \min(\vec{x}_j)\\ \sigma_{ij} &= \sigma_{ij}(\max(\vec{x}_j) - \min(\vec{x}_j))\end{aligned} \tag{20}$$
   e) Set the centers and widths of the new fuzzy sets (in all input dimensions) as in (19)
   f) Set the linear consequent parameter $\hat{\vec{w}}_C$ of the new rule C to $\vec{0}$
   g) Set the inverse Hesse matrix $(R_C^T Q_C R_C)^{-1}$ of the new rule to αI with α a positive big integer.
8) **Else**
   a) Update the p center components of the winner neuron c_{win} by using the update formula in (17) with the choice of an adaptive η as described in (18)
   b) Update the p width components of winner neuron c_{win} by using variance update formula in (16)
   c) Transfer all the cluster centers and widths back to original range as in (20)
   d) Project winner neuron onto the axis to update the antecedent part of the rule belonging to the winner

neuron

e) Correct the linear parameter vector of consequent functions and the inverse Hesse matrix of the to the winner neuron corresponding rule by

$$\hat{\vec{w}}_{win}(k) = \hat{\vec{w}}_{win}(k) + \vec{\delta}_{win}(k+1)$$
$$P_{win}(k) = P_{win}(k) + \Delta_{win}(k+1) \qquad (21)$$

9) Update the ranges of all input and output variables
10) Perform weighted recursive least squares as in (13) and (15) for all C rules, achieving parameter vectors $\hat{\vec{w}}_C(k+1)$ and inverse Hesse matrices $P_C(k+1)$. Here all rules has to be taken into account as in the case if Gaussian fuzzy sets always all rules fire to a certain degree.
11) If the data matrix still contains uncovered data (offline case) or new incoming data points are still available (online case) set $k = k+1$ and goto step 3, otherwise stop

In this form, the incremental learning algorithm is directly applicable, with the exception that one point is still missing: how to estimate the correction vectors and matrices in (21) in order to guarantee stable adaptation. Obviously, these correction terms depend on the change extent of the winner neuron and influences all previously evaluated weighting entries and therefore can be only calculated by reusing the first k points again. This leads to the loss of the benefit to be applicable for fast online identification processes, as the linear parameters have to be re-estimated in each learning step. Nevertheless, a bound to these correction terms can be given, which is not proven here:

Lemma 1: Let $\vec{\delta}_i$ with $i = 1, ...C$ be the correction vector for the independent linear parameter estimations of the C rules, then

$$\lim_{k_i \to \infty} \vec{\delta}_i(k_i) = \vec{0} \qquad (22)$$

i.e. the sequence of correction vectors for all rules tends to 0. The same can be stated for the correction terms of the inverse Hesse matrix. As the correction terms are bounded, it is guaranteed that the algorithm does not 'break out', when setting the correction vectors artificially to $\vec{0}$, which has to be done for empirical tests as they cannot be calculated explicitly. Whenever the if-path in step 7 is entered, indeed, a new rule is set and therefore the fuzzy model extended, but after a while also for this rule the algorithm will converge again. So, when taking into account that at some time instance the whole input range is filled up with data sufficiently enough, hence the complete possible input domain is acquired with fuzzy sets and rules, this leads to the overall convergence of the algorithm. Moreover, the larger k gets for the initial generation of fuzzy models as described in step 2 of algorithm 1, the smaller the correction vector values get for the adaptation process. In this sense, for an online identification system the choice of the initial number of data points is a matter of a tradeoff between stability and an early applicability of fuzzy models generated by the algorithm.

When applying *FLEXFIS* to the data as visualized in Figure 1 a correct approximation of the data can be obtained, which is shown in Figure 2.

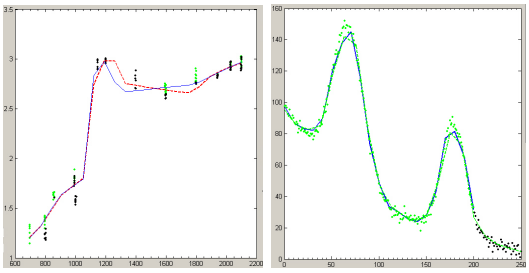


Fig. 2. Comparison of fuzzy models obtained with Algorithm 1 (dotted lines), and *extended genfis2* in closed-loop (solid lines)

TABLE I

COMPARISON OF FUZZY MODEL BUILDING METHODS WITH RESPECT TO QUALITY AND COMPUTATION SPEED

Method	**Quality Training**	**Quality Test**	**Tr. Time online 1 up-to-date every 100 p.**	**Tr. Time online 2 up-to-date each point**
FMCLUST	0.9272	0.849	62m 41s	Not poss.
ANFIS	0.9110	0.842	>genfis2	Not poss.
genfis2 conv.	0.9080	0.810	38m 31s	Not poss.
genfis2 ext.	0.9110	0.844	34m 13s	Not Poss.
genfis2 ext. c. a.	0.8319	0.818	3m 10s	3m 10s
FLEXFIS batch mode 100	0.8712	0.836	4m 36s	Not poss.
FLEXFIS sample mode	0.8411	0.829	10m 57s	10m 57s

VI. EVALUATION ON HIGH DIMENSIONAL DATA

A comparison of the closed-loop as well as open-loop variant of *FLEXFIS* with some widely known closed-loop generation methods of Takagi-Sugeno fuzzy systems will be investigated on high dimensional data sets. This comparison will include the following closed-loop generation methods: *FMCLUST* [3], *ANFIS* [2], [17] and *genfis2 conventional* [14], [15]. Note: with *genfis2 conventional* it is meant the *genfis2* approach as implemented in MATLAB's fuzzy logic toolbox, with *genfis2 extended* the closed-loop variant of *FLEXFIS*, i.e. local estimation of rules' consequents and VQ-ART as clustering method.

In Table I a comparison of these methods based on high dimensional measurement data from an diesel engine recorded at an engine test bench is made. The comparison includes average model qualities of 62 trained up-to-5-dimensional MISO (i.e. Multiple Input, Single Output) fuzzy models on the training data set containing 1810 samples as well as a complete fresh fault-free test data set containing 136 samples. Originally the training data set as well as the test data set contained 80 measurement channels, where 18 channels were completely neglected as input and target channels due to missing data or containing too many outliers. For each of the remaining channels, a fuzzy model was tried to be built up by a subset of the others, which was selected by applying a variable selection technique described in [18]. An overall *r-squared-adjusted* which averages all the *r-squared-adjusted* values of all fuzzy models gives the overall qualities of the methods. Usually, closed-loop modelling methods achieve higher average *r-*

TABLE II

MODEL ACCURACY OF DIFFERENT METHODS WHEN APPLYING TO *auto-mpg* DATA FROM UCI REPOSITORY

Method	**Corr. Coeff.**
FMCLUST	0.917
ANFIS	0.730
genfis2 conv.	0.855
genfis2 ext.	0.916
genfis2 ext. conv. adapt	0.210
FLEXFIS sample mode	0.912

squared-adjusted values than open-loop methods, which is underlined in Table I. This is because closed-loop methods always get in the complete information about the underlying dependencies to be approximated as all data points are sent into the algorithm. However, the obtained qualities on the training set of the with *FLEXFIS* adapted fuzzy models is surprisingly good, as it triggers only a 4.4% decrease when applying batch mode adaptation, respectively a 7.7% decrease when applying sample mode adaptation instead of closed-loop modelling. Moreover, for the qualities on a fresh test set (third column), adaptation approaches can even compete with generation approaches. Column 4 demonstrates clearly, that when models are demanded to be up-to-date after each newly recorded 100 points, all the closed-loop modelling methods take an almost 10 times higher computation speed than *FLEXFIS* in batch mode adaptation. This makes them hardly applicable in fast identification processes and even totally inapplicable for online identification tasks, where fuzzy models have to be up-to-date for each single point (column 5).

Comparing *FLEXFIS* in sample mode with the sample adaptation of the rules consequent parameters alone, denoted as '*genfis2 extended*, c. a.' in Table I, it has to be realized that especially for this data set the adaptation of consequent parameters is sufficient, as the quality of the models do not suffer significantly; moreover, it is three times faster than *FLEXFIS* in sample mode. The reason for this lies in the good distribution and density of the first k data points sent into initial model training. This circumstance can immediately change, when applying a different measurement plan, where the measurements are recorded in a different order. For instance, data from a BMW-diesel engine were recorded in an ascending order with respect to the two main influencing channels at an engine test bench, namely rotation speed and torque. In this case it turned out that the sample mode adaptation of the rules consequent parameters alone is useless, as it delivered an average model quality over 32 models of only 0.67 on fresh test data.

In table II model quality results on the *auto-mpg* data from the UCI-repository[1] are shown. The data concerns city-cycle fuel consumption in miles per gallon and consists of eight attributes including one class attribute (target channel), namely the so-called 'miles per gallon', and seven input attributes, Five input dimensions were enough to describe the relationship significantly, the results are shown in Table II: *FMCLUST*, *genfis2 extended* and *FLEXFIS* show almost similar results with respect to approximation accuracy on fresh test and are even slightly better than the best method demonstrated in [19] (*LAPOC-VS*), which achieves a correlation coefficient of between 0.89 and 0.90. *ANFIS* and *genfis2 extended* with conventional adaptation are more or less forgettable.

[1]http://www.ics.uci.edu/ mlearn/MLRepository.html

VII. CONCLUSION

Concluding, in this paper a new method (*FLEXFIS*) was introduced which is feasible to train Takagi-Sugeno fuzzy models in offline as well as online mode. The applicability and performance of this method in industrial processes was demonstrated due to high dimensional data, where the evaluation results were compared with those of closed-loop approaches for training Takagi-Sugeno fuzzy systems due to approximation accuracy and computational effort. The results of *FLEXFIS* were pretty good as almost the same model qualities could be achieved in comparison with the closed-loop modelling methods, whereas *FLEXFIS* possesses an incremental learning mechanism, which makes it possible to update the models during an online process whenever they have to.

REFERENCES

[1] L. Ljung, *System Identification: Theory for the User*. Upper Saddle River, New Jersey 07458: Prentice Hall PTR, Prentic Hall Inc., 1999.

[2] J.-S. Jang, "ANFIS: Adaptive-network-based fuzzy inference systems," *IEEE Trans. Syst. Man Cybern.*, vol. 23, pp. 665–685, 1993.

[3] R. Babuska, *Fuzzy Modeling for Control*. Boston: Kluwer Academic Publishers, 1998.

[4] E. Bertolissi, M. Birattari, G. Bontempi, and A. D. H. Bersini, "Data-driven techniques for direct adaptive control: the lazy and the fuzzy approaches," *Fuzzy Sets and Systems*, vol. 128, pp. 3–14, 2002.

[5] P. Angelov, *Evolving Rule-Based Models*. Germany: Springer Verlag Berlin, 2002.

[6] P. Angelov and D. Filev, "An approach to online identification of takagi-sugeno fuzzy models," *IEEE Trans. on Fuzzy Systems*, vol. 34, no. 1, pp. 484–498, 2004.

[7] O. Nelles, *Nonlinear System Identification*. Germany: Springer Verlag Berlin, 2001.

[8] O. Nelles and M. Fischer, "Local linear model trees for nonlinear system identification of a cooling blast," in *Proceedings of EUFIT*, 1996.

[9] D. Aha, "Special issue on lazy learning," *Artificial Intelligence Review*, vol. 11, pp. 1–6, 1997.

[10] J.-S. Jang and C.-T. Sun, "Functional equivalence between radial basis function networks and fuzzy inference systems," *IEEE Trans. on Neural Networks*, vol. 4, pp. 156–159, 1993.

[11] L. Koczy, D.Tikk, and T. Gedeon, "On functional equivalence of certain fuzzy controllers and rbf type approximation schemes," *International Journal of Fuzzy Systems*, 2000.

[12] K. Aström and B. Wittenmark, *Adaptive Control - Second Edition*. Addison-Wesley ISBN-0-201-55866-1, 1995.

[13] S. Qin, W. Li, and H. Yue, "Recursive pca for adaptive process monitoring," *Journal of Process Control*, vol. 10, pp. 471–486, 2000.

[14] S. Chiu, "Fuzzy model identification based on cluster estimation," *Journal of Intelligent and Fuzzy Systems*, vol. 2, no. 3, 1994.

[15] R. Yager and D. Filev, "Generation of fuzzy rules by mountain clustering," Machine Intelligence Institute, Iona College, New Rochelle, NY 10801, Tech. Rep. MII-1318R, 1994.

[16] E. Lughofer and E. Klement, "Premise parameter estimation and adaptation in fuzzy systems with open-loop clustering methods," in *Proceedings of FUZZ-IEEE 2004*, Budapest, Hungary, 2004.

[17] J.-S. Jang, "Self-learning fuzzy controllers based on temporal backpropagation," *IEEE Trans. on Neural Networks*, vol. 3, pp. 714–721, 1992.

[18] W. Groißböck, E. Lughofer, and E. Klement, "A comparison of variable selection methods with the main focus on orthogonalization," in *Soft Methodology and Random Information Systems*, ser. Advances in Soft Computing, M. Lopéz-Díaz, M. Gil, P. Grzegorzewski, O. Hryniewicz, and J. Lawry, Eds. Berlin, Heidelberg: Springer, 2004, pp. 479–486.

[19] J. Himmelbauer and M. Drobics, "Regularized numerical optimization of fuzzy rule bases," in *Proceedings of FUZZ-IEEE 2004*, Budapest, Hungary, 2004.

Some generalizations of the pseudo-Laplace transform

Ivana Štajner-Papuga

Department of Mathematics and Informatics, University of Novi Sad
Trg D. Obradovica 4, 21 000 Novi Sad, Serbia and Montenegro
e-mail: stajner@im.ns.ac.yu

1 Introduction

Topic of this paper belongs to the area of mathematics known as pseudo-analysis. Pseudo-analysis is considered to be generalization of the classical analysis, and it has proved itself as a useful tool for solving problems in different aspects of mathematics as well as in various practical problems ([4, 5, 7, 8, 13, 14]). This paper presents some generalizations of the pseudo-Laplace transform, transform that has essential role in dealing with differential or integral equations. Also, the generalization of the exchange formula that transforms pseudo-convolution, generalization of the classical convolution that has applications in probabilistic metric spaces, information theory, fuzzy numbers, optimization, etc., into the pseudo-product will be considered.

A short overview of the basic notions, as pseudo-operations, pseudo-integral and pseudo-convolution, will be given in the Section 2. Pseudo-Laplace type transforms, namely the generalized $(\oplus, \odot)$-Laplace transform and the distorted generalized $(\oplus, \odot)$-Laplace transform, which is generalization done in the style of Moynihan (see [6]), will be given in the Section 3. The corresponding exchange formulas will be considered in the Section 4.

2 Preliminary notions

Let $[a, b]$ be closed subinterval of $[-\infty, +\infty]$ (in some cases semiclosed subintervals will be considered) and let $\preceq$ be total order on $[a, b]$. The structure $([a, b], \oplus, \odot)$ is called a *semiring* if following hold:

- $\oplus$ is *pseudo-addition*, i.e., a function $\oplus : [a, b] \times [a, b] \to [a, b]$ which is commutative, non-decreasing (with respect to $\preceq$), associative and with a zero element denoted by $\mathbf{0}$;
- $\odot$ is *pseudo-multiplication*, i.e., a function $\odot : [a, b] \times [a, b] \to [a, b]$ which is commutative, positively non-decreasing ($x \preceq y$ implies $x \odot z \preceq y \odot z$,

$z \in [a,b]_+ = \{x : x \in [a,b],\ \mathbf{0} \preceq x\})$, associative and for which exists a unit element denoted by $\mathbf{1}$;

- $\mathbf{0} \odot x = \mathbf{0}$;
- $x \odot (y \oplus z) = (x \odot y) \oplus (x \odot z)$.

There are three basic classes of semirings with the continuous (up to some points) pseudo-operations. The first class consists of semirings with idempotent pseudo-addition and non idempotent pseudo-multiplication. Semirings with strict pseudo-operations defined by the monotone and continuous generator function belong to the second class, and the third class contains semirings with both idempotent operations. More on this structure can be found in [4, 5, 7, 10, 11, 12].

Counterpart of the classical measure based on the semiring $([a,b], \oplus, \odot)$ is known as $\sigma - \oplus -$*decomposable measure* (see [5, 7, 8, 10, **?**]). This set function m maps some $\sigma-$algebra Σ into $[a,b]_+$, fulfills conditions $m(\emptyset) = \mathbf{0}$ and

$$m(\bigcup_{i=1}^{\infty} A_i) = \bigoplus_{i=1}^{\infty} m(A_i)$$

for all sequences $\{A_i\}$ of pairwise disjoint sets from Σ, where $\bigoplus_{i=1}^{\infty} x_i = \lim_{n\to\infty} \bigoplus_{i=1}^{n} x_i$. If $\oplus$ is idempotent operation, condition $m(\emptyset) = \mathbf{0}$ and pairwise disjointness of sets can be omitted.

The construction of *pseudo-integral* on the semiring $([a,b], \oplus, \odot)$ with respect to the $\sigma - \oplus -$ decomposable measure m is similar to the construction of the classical Lebesgue integral (see [3, 5, 7, 10]). This integral is denoted with $\int_X^{\oplus} f \odot dm$ for all bounded measurable function $f : X \to [a,b]$.

Another notion necessary for this paper is notion of the generalized pseudo-convolution (see [3, 10, 12]). Let G be subset of $\mathbb{R}$, $*$ a commutative binary operation on $\mathbb{R}$ that fulfills cancellation law and $(G, *)$ semigroup with unit element e. Let $G_+ = \{x | x \in G, x \geq e\}$.

The *generalized pseudo-convolution of the first type* of two functions $f : G_+ \to [a,b]$ and $h : G_+ \to [a,b]$ with respect to a σ-$\oplus$-decomposable measure m is mapping $f \star h : G_+ \to [a,b]$ given by

$$f \star h(x) = \int_{G_+^x}^{\oplus} f(u) \odot dm_h(v), \tag{1}$$

where G_+^x is set of all $u, v \in G_+$ such that $u * v = x$, $m_h(A) = \sup_{x\in A} h(x)$ for $\oplus = \max$, $m_h(A) = \inf_{x\in A} h(x)$ for $\oplus = \min$, and if $\oplus$ has an additive

generator g, then $dm_h = h \odot d(g^{-1} \circ \lambda)$, where $\lambda = g \circ m$ is the Lebesgue measure (generalized g-convolution of the first type).

Generalized pseudo-convolution of the second type is obtained when $(G, *)$ is a group and the pseudo-integral in (1) is taken over the whole set G (see [12]), i.e., the generalized pseudo-convolution of the second type of functions $f, h : G \to [a, b]$ is

$$f \star h(x) = \int_G^{\oplus} f(x * (-t)) \odot dm_h(t),$$

where $(-t)$ is unique inverse element for t.

3 Generalizations of pseudo-Laplace transform

Through this section, semirings of the first and the second class with generated pseudo-multiplication will be considered. However, the first step is to introduce operation on domain of the functions on which pseudo-Laplace type transforms will be applied.

Let L be a binary operation on $[0, +\infty)$ which is non-decreasing in both coordinate, continuous on $[0, +\infty)^2$, commutative, associative, has 0 as identity and fulfills cancellation law. This operation is strict operation and it can be represented by means of generating function (see [1, 3, 11]). Let $l : [0, \infty) \to [0, 1]$ be *multiplicative generator* for binary operation L, that is l is continuous, decreasing function such that $l(0) = 1$ and

$$L(x, y) = l^{-1}\left(l(x)l(y)\right).$$

We shall consider another binary operation $\diamondsuit$ on $[0, \infty)$ that has to be distributive with respect to L, i.e. $L(x, y)\diamondsuit z = L\left(x\diamondsuit z, y\diamondsuit z\right)$. If L is given by the multiplicative generator, operation $\diamondsuit$ can have only the following form

$$x\diamondsuit y = l^{-1}\left(\exp\left(-\ln l(x) \ln l(y)\right)\right).$$

Now, let $([a, b], \oplus, \odot)$ be a semiring from the first or second class. If $([a, b], \oplus, \odot)$ belongs to the first class of semirings, let us suppose that pseudo-multiplication has been given by multiplicative generator θ, i.e. $x \odot y = \theta^{(-1)}\left(\theta(x)\theta(y)\right)$, where $\theta : [a, b] \to [0, 1]$ is continuous strictly monotone function such that $\theta(\mathbf{1}) = 1$, and $\theta^{(-1)}$ is pseudo-inverse function (see [3]). If the semiring in question belongs to the second class, i.e. both pseudo-operations are strict, θ is additive generator for $\oplus$ and multiplicative generator for $\odot$ and $\theta^{(-1)} = \theta^{-1}$.

3.1 Generalized $(\oplus, \odot)$-Laplace transform

Having in mind previously described operation L on domain of some measurable function $F : [0, \infty) \to [a, b]$, as well as properties of pseudo-operations from

semiring $([a,b],\oplus,\odot)$, following generalization of the pseudo-Laplace transform from [11] can be given.

Definition 1 *The* generalized $(\oplus,\odot)$-Laplace transform *of a measurable function* $F:[0,\infty)\to[a,b]$ *is*

$$\mathcal{L}^{\oplus}_{\odot,L}F(z)=\int_{[0,\infty)}^{\oplus}\theta^{(-1)}\left(l(x\diamondsuit z)\right)\odot dm_F(x). \tag{2}$$

Example 2 Let $\oplus=\max$ and let $\odot$ be an Archimedean t-norm T given by continuous and increasing generating function $\theta:[0,1]\to[0,1]$ with $\theta(1)=1$ (see [3]). Generalized $(\max,T)$ -Laplace transform is mapping $\mathcal{L}^{\max}_{T,L}$ defined for all $F:[0,\infty)\to[0,1]$ as

$$\mathcal{L}^{\max}_{T,L}F(z)=\theta^{(-1)}\left(\sup_{x\geq 0}l(x\diamondsuit z)\theta\left(F(x)\right)\right),\qquad z\geq 0.$$

Example 3 Let $([a,b],\oplus,\odot)$ be semiring from the second class given by generating function θ, i.e. $x\oplus y=\theta^{-1}\left(\theta(x)+\theta(y)\right)$ and $x\odot y=\theta^{-1}\left(\theta(x)\theta(y)\right)$. Generalized $(\oplus,\odot)$ -Laplace transform is mapping $\mathcal{L}^{\oplus}_{\odot,L}$ defined for $F:[0,\infty)\to[a,b]$ as

$$\mathcal{L}^{\oplus}_{\odot,L}F(z)=\theta^{-1}\left(\int_{[0,\infty)}l(x\diamondsuit z)\theta\left(F(x)\right)\right)dx.$$

Remark 4 If pseudo-multiplication is an idempotent operation, i.e., semiring in question belongs to the thired class, generalized $(\oplus,\odot)$-Laplace transform coincides with pseudo-Laplace transform from [11]:

$$\mathcal{L}^{\max}_{\min,L}F(z)=\sup_{x\geq 0}\left(\min\left(z,F(x)\right)\right)\qquad\text{and}\qquad\mathcal{L}^{\min}_{\max,L}F(z)=\inf_{x\geq 0}\left(\max\left(z,F(x)\right)\right).$$

Remark 5 Another type of Laplace transform of a measurable function $f:[0,\infty)\to[0,1]$, the (S,T)-Laplace transform, where $([0,1],S,T)$ is the conditionally distributive semiring has been introduced in [3].

3.2 Distorted generalized $(\oplus,\odot)$-Laplace transform

Through this section, a modification of the previously described generalized $(\oplus,\odot)$-Laplace transform, so called *distorted generalized* $(\oplus,\odot)$*-Laplace transform,* shall be considered. This modification is done in the style of Product-conjugate transform used by Moynihan ([6]).

Now, let $([0,1],\oplus,\odot)$ be a semiring of the first or the second class.

Definition 6 *The* distorted generalized $(\oplus,\odot)$-Laplace transform *of a measurable function* $F:[0,\infty)\to[0,1]$ *is*

$$\mathcal{DL}^{\oplus}_{\odot,L}F(z)=\int_{[0,\infty)}^{\oplus}l(x\diamondsuit z)\odot dm_F(x). \tag{3}$$

Example 7 Let us consider semiring of the first class $([0,1],\max,T)$ where T is an Archimedean t-norm. Distorted generalized $(\max,T)$ -Laplace transform is mapping $\mathcal{DL}^{\max}_{T,L}$ defined for $F\in\triangle^+$, where $\triangle^+$ is the space of probability distribution functions, as

$$\mathcal{DL}^{\max}_{T,L}F(z)=\max\left\{0,\sup_{x>0}l(x\diamondsuit z)\theta\left(F(x)\right)\right\},\qquad z\geq 0.$$

Example 8 For $L=+$, distorted generalized $(\max,T_P)$ -Laplace transform has the following form that coincides with Product-conjugate transform ([6]):

$$\mathcal{DL}^{\max}_{T_P,+}F(z)=\sup_{x\geq 0}e^{-xz}F(x).$$

Remark 9 Presented distorted generalized $(\oplus,\odot)$ -Laplace transform, for $\oplus=\max$ and $\odot=T$, has essential role in proving existence of a non-trivial limit for the sequence $\left(\mathcal{T}_{T,L}\left(F_1,F_2,\ldots,F_n\right)\right)_{n\in\mathbb{N}}$, where F_i, $i\in\mathbb{N}$, are probability distribution and $\tau_{T,L}(F,H)(x)$ is well known triangle function. If written in the sense of pseudo-analysis, sequence in question has the following form

$$\left(F_1\star F_2\star\cdots\star F_n\right)_{n\in\mathbb{N}},$$

where $\star$ is generalized pseudo-convolution based on semiring $([0,1],\max,T)$.

4 Generalized pseudo-exchange formula

Let $\star$ be the generalized pseudo-convolution defined by (1) on the semiring $([a,b],\oplus,\odot)$ for $*=L$. It can be proved that the pseudo-exchange for generalized $(\oplus,\odot)$-Laplace transform given by (3) with respect to the corresponding pseudo-convolution $\star$ holds.

Theorem 10 *Let $([a,b],\oplus,\odot)$ be a semiring from the first or the second class, $\mathcal{L}^{\oplus}_{\odot,L}$ corresponding transform given by (3) and $F_1,F_2:[0,\infty)\to[a,b]$ some measurable functions. Then, following holds*

$$\mathcal{L}^{\oplus}_{\odot,L}\left(F_1\star F_2\right)(z)=\mathcal{L}^{\oplus}_{\odot,L}F_1(z)\odot\mathcal{L}^{\oplus}_{\odot,L}F_2(z).$$

Specially, for $L=+$ generalization of Laplace transform from [11] can be obtained. In this case, the pseudo-exchange formula in cooperation with the inverse pseudo-Laplace transform has been applied to determining extremes values of utility functions ([3, 11]).
Exchange-type formula for distorted generalized $(\oplus,\odot)$ -Laplace transform is given by following:

$$\theta^{-1}\left(\mathcal{DL}^{\oplus}_{\odot,L}\left(F_1\star F_2\right)(z)\right)=\theta^{-1}\left(\mathcal{DL}^{\oplus}_{\odot,L}F_1(z)\right)\odot\theta^{-1}\left(\mathcal{DL}^{\oplus}_{\odot,L}F_2(z)\right).$$

Specially, for $\oplus=\max$ and $\odot=T$ we have

$$\mathcal{DL}^{\max}_{T,L}\left(F_1\star F_2\right)(z)=\max\left\{\theta(0),\mathcal{DL}^{\max}_{T,L}F_1(z)\cdot\mathcal{DL}^{\max}_{T,L}F_2(z)\right\}.$$

5 Conclusion

Generalization of the Laplace transform known as pseudo-Laplace transform (see [9, 11]) has proved itself to be useful in solving some nonlinear equations ([5, 7, 8, 9]) as well as in optimization theory and decision theory ([2]). The main aim of this paper has been to present further steps in this process of generalization that could broaden the area of applications. Another problem concerning this topic that should be addressed in the future is existence of inverse generalized $(\oplus, \odot)$-Laplace transform and its possible applications.

Acknowledgement. The work on this paper has been supported by project MNTRS 1866 and network CEEPUS SK-42.

References

[1] J. Aczel, "Lectures on Functional Equations and their Applications", Academic Press, New York, 1966.

[2] F. Baccelli, G. Cohen, G.J. Olsder and J.-P. Quadrat, "Synchronization and Linearity: an Algebra for Discrete Event Systems", John-Wiley and Sons, New York, 1992.

[3] E.P. Klement, R. Mesiar and E. Pap, "Triangular Norms", Kluwer Academic Publishers, Dordrecht, 2000.

[4] V. N. Kolokoltsov and V.P. Maslov, "Idempotent Analysis and Its Applications", Kluwer Academic Publishers, Dordrecht, 1997.

[5] V.P. Maslov and S.N. Samborskij, "Idempotent Analysis", AMS, Providence, 1992.

[6] R. Moynihan, Conjugate transforms and limit theorems for $\mathcal{T}_T$ semigrups, Studia Mathematica 69 (1980/81), 1-18.

[7] E.Pap, "Null-Additive Set Functions", Kluwer Academic Publishers, Dordrecht- Boston-London, 1995.

[8] E. Pap, Decomposable measures and nonlinear equations, Fuzzy Sets and Systems 92 (1997) 205-222.

[9] E. Pap, Applications of decomposable measures, in Handbook Mathematics of Fuzzy Sets-Logic, Topology and Measure Theory (Ed. U. Höhle, R.S. Rodabaugh), Kluwer Academic Publishers, 1999, 675-700.

[10] E. Pap, Pseudo-additive measures and their applications, in Handbook of Measure Theory (Ed. E. Pap), Volume II, Elsevier, North-Holland, 2002, 1403-1465.

[11] E. Pap and N. Ralević, Pseudo-Laplace Transform, Nonlinear Analysis 33 (1998), 533-550.

[12] E. Pap and I. Štajner, Generalized pseudo-convolution in the theory of probabilistic metric spaces, information, fuzzy numbers, optimization, system theory, Fuzzy Sets and Systems 102 (1999), 393-415.

[13] E.Pap, D.Vivona, Noncommutative Pseudo-analysis and its Applications on Nonlinear Partial Differential Equations, J. Math. Anal. Appl. 274/2 (2000), 390-408.

[14] E. Pap, D. Vivona, I. Štajner-Papuga, On a pair of generated pseudo-operations with three parameters and its application on the Burgers type equations, Atti del Seminario Matematico e Fisico dellUniversita di Modena 50 (2002), 451-464.

Quantitative Analysis of Microarray Images

Leila Muresan, Bettina Heise, Jan Kybic, Erich Peter Klement

***Abstract*—**The goal of quantitative analysis of microarrays is to determine the strength of the hybridization for every element of the array (spot). Since new technology allows the detection of the signal at single molecule level, new methods for analysis are necessary. A detection error of 10% is considered acceptable. In this paper we discuss three approaches to single peak detection inside the spots of the arrays, and compare the results we obtain on simulated and real images. These approaches are: global thresholding, an adaptive filter combined with local thresholding, and the third algorithm, we propose, is a statistical background estimation based method combined with clustering, which produces comparable results to well known algorithms, without having to perform the manual adjustment of the parameters.

***Keywords*—**Peak detection, local threshold, adaptive filter**,** noise, statistical outlier detection, clustering

I. Introduction

From an image processing point of view, the task of quantitative microarray analysis consists of recognizing and counting single peaks in fluorescence images. (According to biological terminology, the elements of the microarray are called *spots*, and the bright signals inside them *peaks.* Peaks are diffraction limited point-like objects). Due to the development of an ultra-sensitive microarray platform, based on a combination of anti-adsorptive thin glass slides and the Cytoscout® ([3], [6]) detection at single molecule level becomes possible. At this level, the current techniques [4], based on the computation of the mean intensity for an array element, are obsolete. More sensitive and accurate methods are required.

Among the challenges of this task we identify the robustness to noise, to different concentration of oligonucleotides (resulting in different densities of the peaks to be detected) and the total automation of the procedure. To one pixel of a commercial scanner correspond 400 pixels for the new technique. This leads to a considerable increase in the size of the images that have to be analyzed (the size of one image is approximately 8GB). The size of the images and the high density of peaks makes the peak counting task practically impossible for a human operator. Also, an efficient automatized method should not be computationally too expensive. The results are tested on real as well as simulated data, since ground truth for this kind of images is not available.

As real test images we used 16-bit scans of arrays hybridized with different concentrations of oligonucleotides (0.08, 0.8 and 8 amol/80 µl) resulting in different peak densities in the peak.

For simulations, we generated images containing various numbers of peaks, and added a percentage of Gaussian and/or Poisson noise.

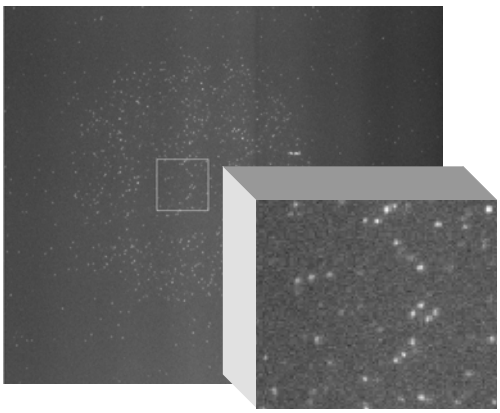


Fig 1. Oligonucleotide microarray peak with single molecule sensitivity (concentration: 0.8 amol/ 80 µl)

A pre-processing step in the scan analysis consists in detecting a region of interest, around the position of every single array element. Each such region will be analyzed separately.

II. Peak detection by thresholding

As a first approach, a global threshold method was used to convert greyscale images into binary images, and count subsequently the resulting blobs. As all the microarray images in our test dataset show a Poisson-like histogram, the triangle algorithm [9] seems suitable for a fast automated thresholding.

However, as a drawback of all global threshold methods, changing illumination profile causes background variations, which result in an overlap of the intensity distributions for background and foreground, making accurate peak detection impossible. Noise may further deteriorate the results.

In order to minimize these effects, in the second approach, we combine a local thresholding method with a previously applied local adaptive smoothing filter [5]. This filter reduces the noise in the original image $x_O(i, j)$ without affecting the

Manuscript received November 5, 2004. This work was supported by the Austrian Genome Program Gen-Au and CEEPUS SK-42.

L.Muresan, B. Heise and E.P. Klement are with the Department of Knowledge-based Mathematical Systems, Johannes Kepler University , Linz, Austria (e-mail: leila.muresan@jku.at, bettina.heise@jku.at, ep.klement@jku.at)

J. Kybic is with Center for Machine Perception, Czech Technical University, Prague, Czech Republic.

peaks, as described by Eq.1,

$$x_{AF}(i,j) = m_l(i,j) \quad + \quad \frac{\sigma_l^2(i,j)}{\sigma_l^2(i,j)+\sigma_n^2}[x_O(i,j) - m_l(i,j)] \qquad (1.)$$

where m_l is the mean intensity value in a local neighbourhood l of pixel (i, j), the variance in the same neighbourhood is denoted σ_l^2, while σ_n^2 denotes the variance of the noise, estimated by $\sigma_n^2 = \frac{1}{N\,M}\sum_{i,j}\sigma_l(i,j)^2$, the mean of all local variances in the image.

The original image $x_O(i, j)$ as well as the filtered image $x_{AF}(i, j)$ are binarized, according to the local adaptive threshold [8], as written in Eq. 2 :

$$b(i,j) = \begin{cases} 1, & for \quad x(i,j) \geq m_l(i,j) + k\,\sigma_n \\ 0, & otherwise \end{cases} \qquad (2.)$$

where k is a constant and is adapted to the imaging conditions.

Finally, pixels, which are found in both the original image x_O and the filtered image x_{AF} by the local threshold, and which are additionally local maxima in both images, are regarded as valid peaks (Fig. 2). The results for different concentrations are summarized in Table II.*A*. Although the results of the local adaptive threshold are acceptable and the method has relatively low complexity, an appropriate estimation of the factor k is necessary, causing additional adaptation efforts for changing concentration and SNR. For this reason, the use of statistical methods for background estimation is justified.

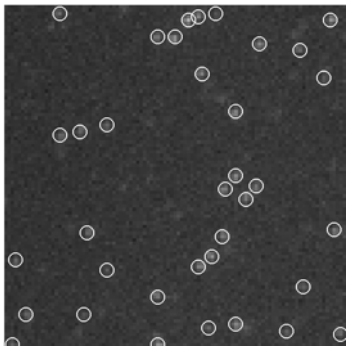


Fig 2. Results of the adaptive filter and local thresholding method for the detail image in fig.1 (k = 1.5, window size = 5)

III. PEAK DETECTION BASED ON STATISTICAL BACKGROUND ESTIMATION

The basic idea is to compute, for each image, certain features, which are sensitive to occurrence of peaks and subsequently, try to find a normally distributed model of the background, for each of these features. The peaks will represent outliers for these normal distributions, and by combination of outliers for different features, we try to eliminate false positives (assuming that noise is uncorrelated among the features). The method is suitable when sufficient background knowledge is available (the peaks represent less than a few percents of the background, which in real images usually is the case).

A. Features for peak detection

The features which are thought to be sensitive to occurrence of peaks are:

1. the variance of a 5 × 5 neighborhood,
2. the mean value of a 3 × 3 neighborhood (after applying the top-hat operation to the image, with a large structuring element, in order to eliminate the background),
3. the discrepancy and modified discrepancy value (described below),
4. the sum of the positive difference between the intensity value of the 4 (or 8 neighborhood) of a pixel and the mean value m of the 5 × 5 neighborhood: $\frac{1}{5}\sum_{|k+l|\leq 1}\left(x(i+k, j+l) - m\right)_+$, where
$$(a)_+ = \begin{cases} a, & if \quad a \geq 0 \\ 0, & otherwise \end{cases} \qquad (3.)$$
5. the result of Laplace filter.

The discrepancy norm $\| x \|_D$ on R^n is defined [2] as a mapping

$$\| x \|_D := \max_{1\leq\alpha\leq\beta\leq n} \left| \sum_{i=\alpha}^{\beta} x_i \right| \qquad (4.)$$

Considering that ideal peaks show a radial symmetric appearance and are more or less size restricted in our images by 5 x 5 pixels, the sequence X of the regarded pixels over which we calculate the discrepancy *D(i, j)* was chosen in the following way:

$$\begin{aligned} X = (x_0, x_1, x_2, \ldots x_{23}) = \\ (x(i,j), x(i+1,j), x(i+2,j), \\ x(i,j), x(i+1,j+1), x(i+2,j+2), \\ \ldots, x(i,j), x(i+1,j-1), x(i+2,j-2)) \end{aligned} \qquad (5.)$$

starting repeatedly in the central point *(i, j)*,. Subtracting the mean value m of X , the modified radial discrepancy D_R. is:

$$D_R = \max_{1\leq\alpha\leq\beta\leq n} \left| \sum_{i=\alpha}^{\beta} (x_i - m) \right|, \qquad (6.)$$

For reduction of computing time only local maxima were considered as central points.

An analysis of "sensitivity" of each feature (the sensitivity to peaks, to the dimension of the chosen neighborhood, to noise) seems to be necessary.

B. Outlier detection

In order to detect outliers in every computed feature, we shall use the "*modified z-score method*", for normal distributions. Let z_{ij} be the value of a feature at pixel *(i,j)* (described in section *A*). The distribution of the feature for the background is standardized, using robust estimators, and then outliers are detected. The method consists of the following steps [7]:

1. Compute the median m of the z_{ij}
2. Compute the median MAD of $\left|z_{ij} - m\right|$
3. If the following inequality holds:

$$0.6745 \cdot \frac{z_{ij} - m}{MAD} > t \qquad (7.)$$

then z_{ij} is an outlier with respect to the feature z.

The effect of the threshold t in Eq. 7 will be discussed below. However one should keep in mind that, in case of normal distribution, 99.7 % of the data is closer to the mean than 3σ, where σ is the standard deviation of the distribution and for outlier detection, usually a value of $t = 3.5$ is chosen.

For variance, the distribution is χ^2 and not the normal one. Either a transformation of the x-axis, or further investigation of closeness to normal distribution are necessary.

C. Counting peaks

Since, in practice, the result still contains a lot of background data, some further steps are necessary. In order to select only the data related to the peaks, clustering algorithms are applied. Since the appearance of the peaks shows a great variability, (due to background variance, focus etc.) it seems more reasonable to divide the candidates, instead of two, into three categories: rejected candidates, weak candidates, strong candidates. We tested two clustering algorithms: fuzzy c-means and Gustafson-Kessel. The Gustafson-Kessel (see [1]) method provides better results, since it is sensitive to the size and form of the clusters.

IV. Results

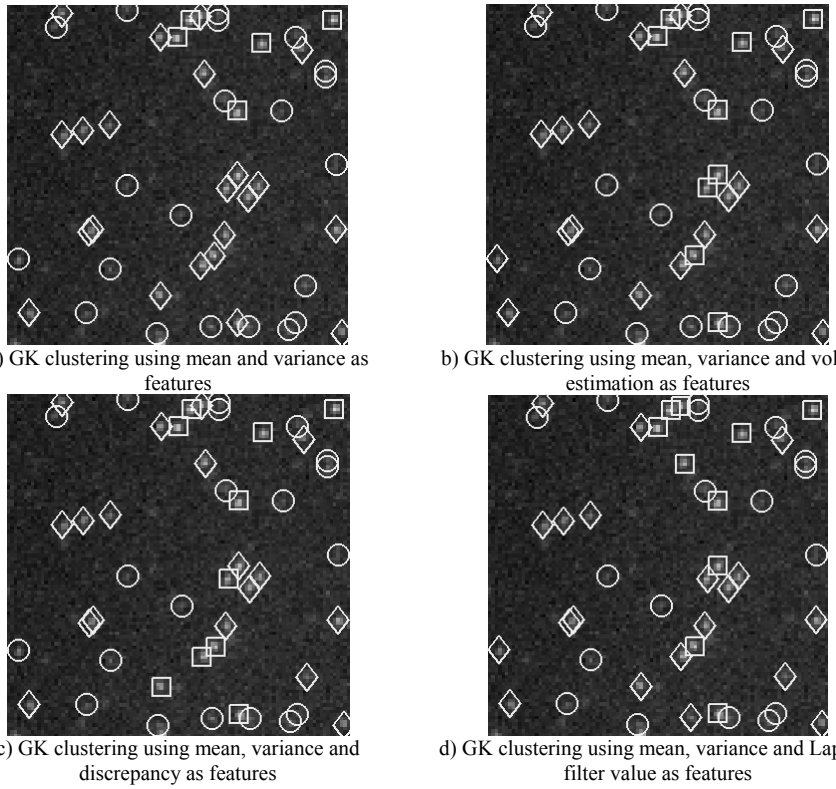
In Table 1, we present the number of peak candidates, found as a result of outlier detection for the features 1 and 2 (variance and mean). The test images contain 100, 500 and 1000 peaks, respectively. The background intensity is a tenth of the mean peak intensity, and Poisson noise was added.

Since the variation is less than 10%, no further processing is necessary.

TABLE I
NUMBER OF PEAKS SELECTED AS OUTLIERS FOR DIFFERENT THRESHOLD VALUES (FEATURES: VARIANCE AND MEAN). THE RESULTS VARY LESS THAN 10%

Generated peaks	Detected peak candidates		
	$t = 3.5$	$t = 3.0$	$t = 2.5$
100	95	99	109
500	503	521	546
1000	961	988	1021

In real images, peaks show a great variability, the features change if the peak is defocused, and the SNR of the recorded images is low. In order to minimize these negative effects, we lower the outlier detection threshold to $t = 3$, introducing some background elements.



a) GK clustering using mean and variance as features

b) GK clustering using mean, variance and volume estimation as features

c) GK clustering using mean, variance and discrepancy as features

d) GK clustering using mean, variance and Laplace filter value as features

Fig 3. Detail of clustering result (GK clustering with different features) for microarray image having concentration 0.8 amol/80 µl. The three resulting clusters are: □ – clear peak, ◊ - peak out of focus, ○ - rejected peak candidate

TABLE II

A) NUMBER OF PEAKS IN MICROARRAY IMAGES USING ADAPTIVE FILTER AND LOCAL THRESHOLDING ($K = 1.5$)

Concentration (amol / 80 µl)	Detected peaks
0.08	101
0.8	875
8	3808

B) NUMBER OF PEAKS IN MICROARRAY USING DIFFERENT FEATURE COMBINATIONS AND GK CLUSTERING (C_1 REJECTED, C_2 DEFOCUSED, C_3 FOCUSED OBJECTS)

Concentration (amol / 80 µl)	Features	Detected peaks					
		$t = 3.5$			$t = 3.0$		
		C_1	C_2	C_3	C_1	C_2	C_3
0.08	1, 2, 3	104	44	27	141	58	30
	1, 2, 4	98	54	23	109	74	46
	1, 2, 5	105	46	24	145	56	28
	1, 2	107	48	20	152	56	21
0.8	1, 2, 3	503	560	158	553	610	175
	1, 2, 4	503	530	188	585	561	192
	1, 2, 5	532	493	196	596	543	199
	1, 2	578	531	112	645	566	127
8	1, 2, 3	4409	1032	1040	4566	1132	1081
	1, 2, 4	3365	1267	1849	3048	2647	1084
	1, 2, 5	3118	2408	955	3218	2573	988
	1, 2	3258	2556	667	3535	2628	616

After performing a clustering in three classes, in order to reflect the characteristics of wide-field fluorescence microscopy imaging, (focused objects, defocused objects and noise), the class of candidates least resembling to the assumed model is discarded as the elements introduced by noise. The weak candidates represent mainly defocused objects, while the best are accepted as valid peaks. The results of the Gustafson-Kessel clustering for three different concentrations are summarized in Table II.*B*. The three numbers represent the cardinality of each cluster (the first is the number of rejected peaks, while the second and the third represent the cluster of defocused and focused peaks, respectively). In the "*Features*" column, the features used for clustering are enumerated (denoted as in III.*A*). The set of peaks corresponding to focused and defocused objects varies little with the features used for clustering, the most significant differences occurring at very high concentrations.

V. CONCLUSIONS

The task of peak counting for very large images, with little signal cannot accommodate human intervention. The last method presented in the paper produces comparable results to well known algorithms, without having to perform the usual adjustment of the parameters. The results of this method can be used in a further analysis of peaks. Nevertheless, additional tests are necessary as well as an analysis of the sensitivity to different peak concentration, SNR and varying imaging conditions, and improvement of the algorithm time-complexity by using a sub-sample of the original data.

ACKNOWLEDGMENT

We would like to thank the members of the Biophysics Institute, at Johannes Kepler University (Linz) and the Upper Austrian Research for interesting discussion and suggestions, and the support of the GEN-AU research program.

REFERENCES

[1] Babuska R.- Fuzzy modeling for Control, Kluwer, 1998

[2] Bauer P., Bodenhofer B., Klement E.P., A fuzzy system for image pixel classification and its genetic optimization. In Trappl R. (Ed.), *Cybernetics and Systems* '96. Austrian Society for Cybernetic Studies, Wien, Vol. 1, pp. 285--290. 1996

[3] Hesse J., Sonnleitner M., Sonnleitner A., Freudenthaler G., Jacak J., Höglinger O., Schindler H., Schütz G.- Single molecule reader for high-throughput bioanalysis, *Anal. Chem.* , 76, pp. 5960-5964, 2004

[4] Jain A. N., Tokuyasu T. A., Snijders A.M., Segraves R., Albertson D, G. Pinkel D.- Fully Automatic Quantification of Microarray Image Data, *Genome Researc*h, Vol. 12, Issue 2, 325-332, 2002

[5] Kuan DT, Sawchuk AA, Strand TC, Chavel P. Adaptive noise smoothing filter for images with signal-dependent noise, *IEEE Transaction on Pattern Analysis and Machine Intelligenc*e, 1985, 7(2), 165-177

[6] Schindler H.. Vorrichtung zur Visualisierung von Molekülen. PCT/AT99/00257, international patent, 2000

[7] Staudte R.G, Sheather S. J. - Robust estimation and testing, Wiley, 1990

[8] Trier OD, Jain AK.. Goal-Directed Evaluation of Binarization Methods, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1995, 17 (12): pp. 1191-1201

[9] Zack G.W., Rogers W.E., Latt S.A.. Automatic Measurement of Sister Chromatid Exchange Frequency. *J. of Histochemistry and Cytochemistry*, 25(7): pp. 741-753, 1977.

pFCSP and Imlementation of Priority Queries in FSQL

Aleksandar Takači

Faculty of Technology, University of Novi Sad
Bulevar Cara Lazara 1, 21000 Novi Sad, Serbia and Montenegro
E-mail: atakaci@uns.ns.ac.yu

Abstract: The concept of priority is often used in real time systems. Priority t-norms are used to capture this concept. System called pFCSP uses priority t-norms. It captures and implements the concept of priority. Using this system we can broaden the query language FSQL with queries that can handle priority. In our model priority reasoning which has theoretical background is incorporated giving us yet another dimension in knowledge acquisition through data mining.

Keywords: priority t-norm, fuzzy relational databases, FSQL, pFCSP

1 Introduction

Priority is most often viewed as the importance level of an object. The concept of priority is often used in real time systems.

FSQL is a query language which implements fuzzy queries and it is an upgrade of standard SQL. Most often it is implements on fuzzy databases and supports fuzzy notions i.e. high, low, large small etc. (see [1, 5])

In this paper a proposition for including priority into FSQL will be given. The theoretic background will be the axiomatic framework of pFCSP's (prioritized fuzzy constraint satisfaction problems). pFCSP's were introduced by Dubois, and an axiomatic framework was given in [4]. pFCSP is actually a FCSP (fuzzy constraint satisfaction problem) in which the notion of priority is introduced. One of the key factors in that implementation are priority t-norms. They are introduced in such a way that the smallest value (usually the value with the biggest priority) has the largest impact on the result given by priority t-norm.

We will give an axiomatic framework for pFCSP and also some systems which satisfy this framework thus justifying its cause.

Finally, we will show what type of queries can be included into FSQL and sketch the implementation idea avoiding technicalities.

2 Priority t-norms

In to order to introduce priority into FCSP (fuzzy constraint satisfaction problem) priority t-norms are used. When it is necessary to satisfy all the constrains a t-norm is used to aggregate the degree of satisfaction of each one of the constrains. PFCSP (prioritized fuzzy constraint satisfaction problem) is designed in such a way that the constraint with the largest priority is more likely to have a smaller value. Definition of priority t-norms is such that the increase of the smallest value (most often the value of the constraint with the highest priority) has a bigger impact i.e. gives larger values of the general satisfaction degree than an increase on any other components. More details on PFCSP, axioms, instantiation and validation is given in [4]. Priority t-norms are defined in the next definition.

Definition 1 *A t-norm is called a priority t-norm if the following condition holds for all $a_1 \leq a_2$ in $[0,1]$ and δ such that $a_2 + \delta \leq 1$:*

$$T(a_1 + \delta, a_2) \geq T(a_1, a_2 + \delta)$$

3 pFCSP - Prioritized Fuzzy Constraint Satisfaction Problem

Definition 2 *A fuzzy constraint satisfaction problem (FCSP) is defined as a 3-tuple (X, D, C^f) where:*

1. *$X = \{x_i | i = 1, 2, \cdots, n\}$ is a set of variables.*
2. *$D = \{d_i | i = 1, 2, \cdots, n\}$ is a finite set of domains. Each domain d_i is a finite set containing the possible values for the corresponding variable x_i in X.*
3. *C^f is a set of fuzzy constrains. That is,*

$$C^f = \{R^f | \mu_{R_i^f} : (\prod_{x_j \in var(R_i^f)} d_j) \Rightarrow [0,1], i = 1,2,\cdots n\}$$

Definition 3 *In a FCSP (X,D,C^f), given a compound label v_x of all variables X, the global satisfaction degree for the compound label v_x is defined as*

$$\alpha(v_X) = \oplus\{\mu_{R_i^f}(v_{var(R^f)}) | R^f \in C^f\}$$

where $\oplus$ is an aggregation operator on the unit interval. A solution of FCSP is a compound label v_X such that

$$\alpha(v_X) \geq \alpha_0$$

where α_0 is called a solution threshold which is usually predetermined.

If all constrains in FCSP have to be satisfied a t-norm is used for $\oplus$. If at least one of the constrains has to be satisfied a t-conorm is used.

In order to introduce the concept of priority into FCSP the following an axiomatic framework is given in [4]. The sytems conists of 5 axioms.

4 FSQL

4.1 Fuzzy Relational Databases and FSQL

In order to expand classical relational databases to model impression fuzzy relational databases (FRDB) are introduced. Many models dating up to 1982 of FRDB have been studied.

Classical databases store only precise information. For example if a persons height is not known but it is known that a person is a tall person classical databases cannot store this information. On the other hand FRDB through the concept of linguistic labels can have the value *tall* for attribute *height*. Besides storing imprecise values FRDB through FSQL can answer a broader set of queries. Take for example queries: *I need students of medium height that have good marks in PE* or *Give me people with average salaries and small housing capacity* etc.

4.2 Incorporating priority queries into FSQL

In the previous section we have introduced pFCSP who can handle priority. Also, the a concrete pFCSP which uses T_L and S_p or any Frank t-norm and t-conorm can easily be implemented. For example, our system can handle calculations of type: Is it better to have fair GPA and excellent physical condition or good GPA and good physical condition if you want to teach PE in a high school. This concept can be easily incorporated into FRDB. For example queries of type:

SELECT * FROM Students
WHERE GPA FEQ Good WITH priority HIGH
AND MathGrade GEQ 7 WITH priority MEDIUM
AND Sex EQ Male WITH priority LOW
SORT DESC

These type of queries add a new dimension to FRDB. They allow us to choose based not only on aggregated values of all attributes but also taking onto account the importance (priority) of a particular attribute. Someone would argue that instead of priority we can use crisp thresholds for each attributes giving higher thresholds to the more important attributes but this would lead to a crisp join of α cuts which could not lead us to the proper answer. In our model priority reasoning which has theoretical background is incorporated giving us yet another dimension in knowledge acquisition through data mining.

There are many technical and implementations details to be done in order for this concept to be implemented but we hope that they will be solved and we will have a pFSQL i.e. FSQL that can handle priority queries.

5 Acknowledgements

This paper was written with the support of the Serbian Ministry of Science and Technology project No. 1866 and CEEPUS SK-42 network.

References

[1] J. Galando, M. C. Aranda, J. L. Caro, A. Guevara, A. Aguayo Applying fuzzy databases and FSQL to the managmnet of rural acomodation, Tourism Managment 24, (2003) 457-463.

[2] E. Klement, R. Mesiar, E. Pap, Triangular norms", Series: Trends in Logic, Kluwer Academic Publishers, , Vol. 8, Dordrecht, 2000.

[3] X. Luo, N. R. Jennigs, N. Shadbolt , H. Leung, J. H. Lee, A fuzzy constraint based model for bilateral multi isssue negotiations in semi competitive enviroments, Artificial Intelligence 148 (2003) 53-102.

[4] X. Luo, J. H. Lee, H. Leung, N. R. Jennigs , Prioritised fuzzy constraint satisfaction problems: axioms, instantiation and validation, Fuzzy sets and systems 136 (2003) 151-188.

[5] H. Prade, C. Testemale, Fuzzy Realtional databases: Representational issues and reduction using similarity measures, Journal of the American Society of Informational Sciences 38(2), (1987), 118-126

[6] R. G. Ricci, M. Navara, Convexity conditions and their additive generators, Fuzzy sets and systems 136 , article in print.

Technical Report
FLLL–TR–0401

FUZZY.LOGIC.LAB.LINZ

The data/view model: application examples

Roland Richter
Fuzzy Logic Laboratorium Linz
email: roland@flll.jku.at

Abstract — The data/view model and the Boost.View library can be used in various areas. In this paper, several possible examples of their application, from simple string parsing to more complex signal and image processing algorithms, are presented.
One topic which receives special attention is image segmentation and enhancement. Starting with a well known classification method, namely the discrepancy norm [BBK96], we develop an "image enhancement view" which applies different filters onto different image areas to remove noise and unwanted artefacts.

Key words — *data/view model, image processing*

Johannes Kepler Universität Linz

Institut für Algebra, Stochastik und
wissensbasierte mathematische Systeme
Johannes Kepler Universität Linz
A-4040 Linz
Austria

Fuzzy Logic Laboratorium Linz-Hagenberg

FLLL
Softwarepark Hagenberg
Hauptstrasse 99
A-4232 Hagenberg
Austria

> In theory there is no difference between
> theory and practice. In practice there is.
> - Yogi Berra

1 String parsing

As a first example, lets look at the processing of strings (which can be “viewed” as a container of characters). Suppose we want to split up a string such as

```
One sentence with five words.
```

into its five single words `One`, `sentence`, `with`, `five`, and `words` – the later without the period. Again, a picture can help to understand our goal – see FIGURE 1. Note that this picture in some sense looks like the “inverse” of the chain view picture.

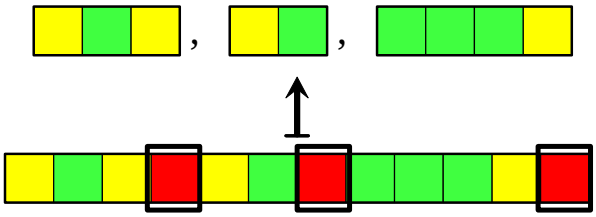


FIGURE 1: Parsing does not modify the containers’ contents, but modifies the structure.

The first trial I undertook some time ago looked like LISTING 1.

```
string s( "One sentence with five words." );

typedef transform_view< string, boost::function1<bool,char> >
        TrueIffBlankView;
TrueIffBlankView view( s, bind2nd( equal_to<char>(), ' ' ) );

TrueIffBlankView::const_iterator b = view.begin();
TrueIffBlankView::const_iterator e;

while( ( e = find( b, view.end(), true ) ) != view.end() )
{
  cout << string( s.begin() + (b-view.begin()),
                  s.begin() + (e-view.begin()) ) << endl;
  b = e + 1;
}
```

LISTING 1: Writes the five words to the screen.

Wait, what’s happening here? We use the transform view to iterate over the string; each time the transform views’ function returns true – which means that the iterator points to a blank in between

words – we send the string formed from between b and e to the standard output. That's not very elegant, and in fact we can go much further than that.

To do so, lets write down what parsing means. Given a string, which we treat as a container of characters, here's the strategy:

1. Place breaking marks at certain locations – for instance, blanks, commas, periods, etc. That's what a filter view can do: given a "is a character indicating a break"-predicate, it only returns (pointers to) those characters.
2. Form new strings which hold exactly the text between two breaking marks. That might be the task of a transform view which is initialized with some string generating function. Since a transform view can only hold a unary function, we have to add another view:
3. Pair together two consecutive (pointers to) breaking marks. Each such pair describes the range of a string, stretching from the pairs' first member up to the pairs' second member. Pairing together might be done with the help of a neighbour view.

Summarizing, we have got to deal with three different views: First, a filter view filters out only those characters which indicate a breaking location. Second, a neighbour view pairs together consecutive such locations together into ranges. Third, a transform view takes such ranges and creates string out of them.

```
struct isBreakingCharacter
  : public std::unary_function< char, bool >
{
  bool operator()( char ch ) const
  {
    return( isalnum( ch ) == 0 );
  }
};
```

LISTING 2: Break if character is not alpha-numeric.

Assuming that we have two functions (or rather function objects), namely, isBreakingCharacter() which tells us whether we should break at a certain character or not (compare LISTING 2, and constructString() which constructs strings out of ranges, we can define the exact view types. Note that LISTING 3 shows very clearly how one view is stacked upon the other.

Using these type definitions, writing the working code is relatively straight-forward. Unfortunately, there's just one further detail which has to be observed: the first breaking character appears somewhere after the beginning of the string. In our case, the first blank is *after* the word "`One`". Consequently, the first pair of consecutive characters subsumes the blank after `One` and that after `sentence`; which means that the first constructed string is "`sentence`" – we missed the first word! As a workaround, lets insert a blank at the front of the sentence; finally, we can parse the string as shown in LISTING 4.

It is noteworthy – and a little bit disappointing – that even this relatively simple task requires not only to stack three different views upon each other, but also some non-intuitive hacks. Partly, this

```
typedef filter_view< string, isBreakingCharacter >
        OnlyCharsAtBreakView;

typedef neighbour_view< OnlyCharsAtBreakView, 2 >
        PairTogetherView;

typedef transform_view< PairTogetherView, constructString >
        ConstructStringsView;
```

LISTING 3: Definition of the three views.

```
std::string s( "One sentence with five words." );

OnlyCharsAtBreakView onlyAtBreak( s );

// HACK Inserts break point at front of s to get the first word as well:
onlyAtBreak.domain().insert( (std::string::size_type)0, 1, ' ' );

PairTogetherView::difference_type consecutivePairs( 0, 1 );
PairTogetherView pairTogether( onlyAtBreak, consecutivePairs );

ConstructStringsView result( pairTogether );
```

LISTING 4: The final code.

is so because views only allow local operations; for algorithms which require non-local data – and parsing is such an operation –, we have to apply certain tricks.

On the other hand, this examples also demonstrates the "pros" of using views: for instance, up to the last line, where the final view result is constructed, we did just that: constructing views. We did *not* have any other computations due to lazy evaluation.

Out of curiosity, I'd like to advance this example a little bit further. Sometimes it is desirable to keep a certain part of the text "as is" although it contains breaking characters. A simple possibility to tell the algorithm that it should left this part unchanged is to surround this part with so called *escape characters* – typically, quotes, brackets and the like. Suppose we have got

```
A sentence "for demonstration purposes only" with nine words.
```

This sentence should be split up into *six* parts: namely, `A`, and `sentence`; the fragment `"for demonstration purposes only"` should be kept as a whole; then, `with`, `nine`, and `words`.

One solution might be to extend function isBreakingCharacter() as shown in LISTING 5: an internal state insideQuotes is added to the function object which keeps records of whether it is currently inside or outside a matching pair of quotes.

One possible drawback of this straight-forward implementation is that it assumes that the string is parsed from left to right. This is a general danger when working with functions that have an internal state. Note however that a filter iterator already is forward only, so this assumption does not impose any further restriction.

```
struct isBreakingCharacter
  : public std::unary_function< char, bool >
{
  isBreakingCharacter()
    : insideQuotes( false )
  { }

  bool operator()( char ch )
  {
    if( ch == '\"' )
    {
      insideQuotes = !insideQuotes;
      return false;
    }
    else
      return( !insideQuotes && ( isalnum( ch ) == 0 ) );
  }

private:
  bool insideQuotes;
};
```

LISTING 5: Breaks at non-alphanumeric characters *outside* quotes.

2 Signal processing

2.1 Sampling

To be able to apply signal processing algorithms, we first need a *discrete* signal. Hence, the starting example to demonstrate the application of views in the signal processing domain is to down-sample a continuous signal into a discrete one.

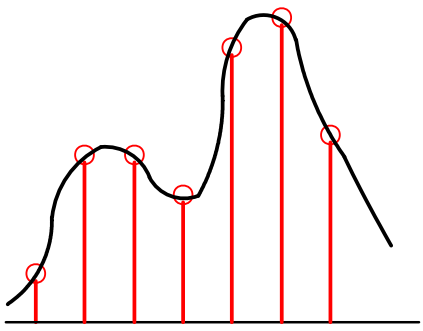


FIGURE 2: Continuous signal is sampled to obtain a discrete one.

Suppose we have a formula which describes the given signal. Sure enough, we can put this generating function into a function_view and already get a discrete signal with step width 1. To sample the signal with another sampling rates, say, with step width 2, we could either

1. manually iterate through the function_view with step 2,
2. use a step_iterator with step width 2 to iterate through the function_view, or
3. wrap a permutation_view with an appropriate re-indexing scheme around the function_view.

The later case is shown in LISTING 6. Note that due to the "laziness" of views, the generating function is not called until the value is actually fetched. Thus, either the sampled view itself can be used, or, as shown in LISTING 6, we decide to copy the sampled signal into another container such as a vector.

```
typedef boost::view::function_view< GenerateSignal > SignalFunction;
typedef boost::view::permutation_view<
          SignalFunction, std::vector<int> > SamplingView;

// Samples in reverse order, and only every second value.
std::vector<int> indices;
for( int i = 0; i < 64; ++i )
  { indices.push_back( 2*(63-i) ); }

SignalFunction signalF( 0, 64 );
SamplingView sampledF( signalF, indices );

std::vector<double> signal( sampledF.size() );
std::copy( sampledF.begin(), sampledF.end(), signal.begin() );
```

LISTING 6: Sampling a continuous signal.

2.2 Windowing

One of the most important tools to analyse a (discrete and periodic) signal is the Discrete Fourier Transform (DFT), which converts the given signal into its frequency domain representation. In order to avoid some unwanted effects such as spectral leakage [Smi97], the signal is often multiplied with a window function before calculating its DFT.

There exists a multitude of different window functions; one of the most popular is the *Hamming window*. Given a discrete signal of length N, the Hamming window is described by the formula

$$w[k] = 0.54 - 0.46 \cos \frac{2\pi k}{N+1} \quad \text{for } k = 0, \ldots, N.$$

We might now implement this operation in terms of the data/view model: first of all, the signal itself is assumed to be stored in a container, such as a STL vector. The Hamming window can be represented by a function view. The element-wise multiplication of the signal with this view requires a transform view with multiplication as its transformation function.

Since this is a binary function, and since a transform view can only operate on a single container with a unary function, another in-between mechanism is necessary which glues together these two containers (or rather, the container and the view) to form another container holding pairs of

```
typedef std::vector<double> Signal;
typedef boost::view::function_view<HammingWindow> HammingView;

typedef boost::tupple::container_tuple<Signal, HammingView> SignalPairs;
typedef boost::view::transform_view<SignalPairs, Multiplication>
  WindowedSignal;
```

LISTING 7: Signal container and window view combined.

elements. Thus, once again, an intermediate view which pairs elements together is necessary; this time, however, pairs do not consist of elements from the same container (as it was the case in the last example), but of corresponding elements of two containers of the same size. As outlined in [Ric02], a container_tuple of size 2 does exactly this, and it does not make any difference that one of its arguments is not a container, but a function view.

Using the two functions (or rather function objects) HammingWindow which calculates the Hamming function, and Multiplication which multiplies the two coefficients of a pair, LISTING 7 shows all types necessary to accomplish our task. Assuming that $N = 64$, these types can then be used to construct the necessary views, as shown in LISTING 8.

```
Signal signal;
// Fill signal with data.

HammingView hamming( 0, 64, HammingWindow( 64 ) );
SignalPairs sigPairs( signal, hamming );
WindowedSignal windowed( sigPairs );
```

LISTING 8: Multiplies a signal with a Hamming window function

3 Image iteration

Iteration is inherently a one-dimensional concept. Returning once again to our famous loop,

```
for( int i = 0; i != N; ++i )
  result[i] = function( source[i] );
```

how can we extend that basic mechanism to two- (or more) dimensional data? Using indices and **operator**[], this seems really quite trivial:

```
for( int j = 0; j != M; ++j )
  for( int i = 0; i != N; ++i )
    result[i][j] = function( source[i][j] );
```

Assuming that an image is stored as a vector of vector of pixels, i.e. as
vector< vector<PixelType> > – which is not a common way of storing images – our loop reads like this:

```
vector< vector<PixelType> >::iterator row;
```

```
vector<PixelType>::iterator col;

for( row = image.begin(); row != image.end(); ++row )
  for( col = row.begin(); col != row.end(); ++col )
    // uses *col
```

Of course, if the image was stored as one large chunk of data in memory – and that *is* most often the case –, one might as well apply apply one-dimensional iteration. However, structure information is lost that way. For instance, if we wanted to process a rectangular section of the image only, this kind of representation would be rather troublesome.

3.1 Two-dimensional iteration

As an example how to implement two-dimensional image iteration, I present the VIGRA image processing software package [Köt]. As described in the manual, and in more detail in [Köt99] and [Köt00], VIGRA employs two-dimensional iteration. As Ullrich Köethe points out this "is not directly possible using operator overloading." Instead, a nested class ImageIterator is created which contains the structures to iterate both in horizontal and vertical direction. This allows to iterate in both directions independently, as shown in LISTING 10.

```
class ImageIterator {
  public:
    // ...

    class MoveX {
        // data necessary to navigate in X direction
      public:
        // navigation function applies to X-coordinate
        void operator++();
        // ...
    };

    class MoveY {
        // data necessary to navigate in Y direction
      public:
        // navigation function applies to Y-coordinate
        void operator++();
        // ...
    };

    MoveX x;    // x-view to navigation data
    MoveY y;    // y-view to navigation data
};
```

LISTING 9: Outline of two-dimensional image iterator. From [Köt00].

```
ImageIterator i(...);
++i.x; // move in x direction
++i.y; // move in y direction
```

LISTING 10: Iteration in two dimensions. From [Köt00].

3.2 Matrix view

In this presentation, I'd like to develop another approach. Remember that a view can change the appearance of a container. So why not looking for a view which attaches a two-dimensional "look" to a one-dimensional container?! More precisely, we strive for a view which wraps a one-dimensional container and provides some kind of iterator that allows two different types of iteration:

- inner (horizontal) iteration: proceeds from one pixel to the next between a given begin/end pair which delimits the current row.
- outer (vertical) iteration: moves the begin/end pair from one row to the next.

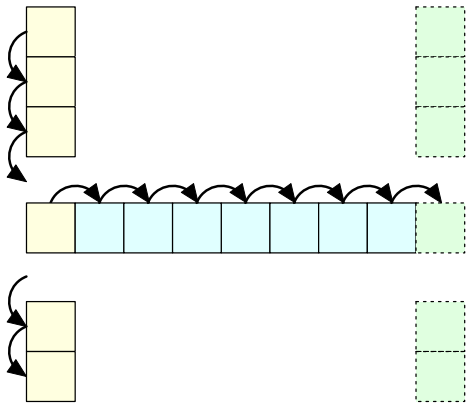


FIGURE 3: Iteration over an image: whereas the outer loop iterates over rows (given as begin/end-pairs), the inner loop iterates over pixels.

Such a structure is what I'd like to call a *matrix view*. FIGURE 3 depicts the situation once again. Translated to code, the two nested loops to iterate over the complete image (or over a rectangular part of it) will read something like this:

```
// Outer loop advances from one row to the next
for( MatrixView::iterator row = view.begin(); row != view.end(); ++row )
{
  // Inner loop advances from one pixel to the next
  for( PixelIterator pixel = row.first(); pixel != row.last(); ++pixel )
  {
    // use *pixel
  }
}
```

4 Image processing and enhancement

Now that we found a suitable way how to iterate over a two-dimensional image, and therefore provide a way how to treat an image as a view, the next question I want to raise is: which image processing algorithms can be re-formulated in terms of the data/view model?

Definitely, not every algorithm can be transformed. As an example, consider the connected components algorithm, which determines the component a pixel is contained in within a (binary) image. This algorithm obviously requires global image information; since views operate on a local level, this algorithm is not suitable for use with the data/view model.

On the other hand, if we restrict ourselves to "local operations", a re-formulation might be possible and useful. For instance, pixel-wise operations are ideally suited for the use of views; such operations could be, for instance,

- colour-to-colour or colour-to-grey scale conversions
- calculation of the negative image
- brightness, contrast, saturation, or colour enhancements.

Also operations which consists essentially of a re-ordering of pixels are candidates for a re-formulation. Typical simple examples are if one wants to

- rotate an image left or right, or
- flip an image vertically or horizontally.

Many image processing algorithms are "local" in the sense that they only require a small neighbourhood around the current pixel to work on. Examples are

- local filter operations such as blur or sharpen
- most edge detection algorithms.

Another such function which is calculated using a 3×3 neighbourhood, and which I'd like to consider in greater detail in the next subsection, is the *discrepancy norm* [BBK96].

One question arises when working with such functions: How to deal with image boundaries? In principle, there are two ways how to proceed at the boundaries where the neighbourhood exceeds the image borders vertically or horizontally:

1. Ignore such pixels completely. For instance, for a 3×3 neighbourhood, all pixels in the top-most and bottom-most row and in the left-most and right-most column would be skipped.
2. Wrap the neighbourhood around the image boundaries.

For the later approach, using a cyclic iterator which wraps around containers might be useful.

4.1 Image segmentation with the discrepancy norm

Among the multitude of algorithms to detect edges in images, there are some approaches that utilize fuzzy reasoning. More precisely, for each pixel in the image some suitable classification numbers are calculated; then, these numbers are used as input values to a fuzzy system which contains the rules describing the "edginess" and which decides whether the pixel is an edge pixel or not.

This mechanism can be either supervised, as described in [Ara00], or unsupervised, as is the approach using the discrepancy norm presented in [BBK96]. This approach has the additional advantage that it does not only separate between edge and non-edge pixels, but does a classification for each pixel whether it lies in a "Homogeneous", "Edge", "Halftone", or "Picture" area.

The approach of [BBK96] is based on the discrepancy norm:

4.1. DEFINITION.

The mapping

$$\begin{aligned} \|.\|_D : \mathbb{R}^n &\longrightarrow \mathbb{R}, \\ \vec{x} &\longmapsto \max_{1 \le a \le b \le n} \left| \sum_{i=a}^{b} x_i \right| \end{aligned}$$

is called the *discrepancy norm* on $\mathbb{R}^n$.

Since this formula would require $\mathcal{O}(n^2)$ operations to calculate, the following formula is more practical to use:

4.2. THEOREM.

Let $X_j := \sum_{i=1}^{j} x_i$ denote the partial sums for $1 \le j \le n$. Then, for all $\vec{x} \in \mathbb{R}^n$,

$$\|\vec{x}\|_D = \max_{1 \le b \le n} X_b - \min_{1 \le a \le n} X_a$$

holds.

PROOF.
See [BBK96]. □

Time for some definitions: we work with either grey level or RGB images of dimension $W \times H$, where each of the three colour bands has 1 byte, i.e. 8 bits, of information. More formally, this reads as:

4.3. DEFINITION.

A $W \times H$ matrix of the form

$$(v(i,j)),\ i = 0, \ldots, W-1,\ j = 0, \ldots, H-1$$

is called an 8 bit grey level image of width W and height H. Its entries $v(i,j) \in \{0, \ldots, 255\}$ are called pixels at (i,j).
A $W \times H$ matrix of the form

$$((r(i,j), g(i,j), b(i,j))),\ i = 0, \ldots, W-1,\ j = 0, \ldots, H-1$$

is called a 24 bit RGB colour image of width W and height H. Its entries

$$p(i,j) := (r(i,j), g(i,j), b(i,j)) \in \{0, \ldots, 255\}^3$$

are called RGB pixels at (i,j), where $r(i,j)$ represents the red, $g(i,j)$ the green, and $b(i,j)$ the blue portion of the pixel.

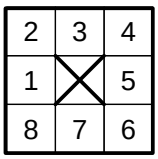


FIGURE 4: Enumeration of pixels within the 3×3 neighbourhood.

In order to calculate the discrepancy norm for a given 3×3 neighbourhood of a pixel, the neighbouring pixels have to be enumerated such that they form a tuple of size eight. This is done as shown in FIGURE 4, that is, we define an enumeration mapping $l_{i,j} : \{1, \ldots, 8\} \longrightarrow 0, \ldots, W-1 \times 0, \ldots, H-1$ which maps $1 \mapsto (i, j-1)$, $2 \mapsto (i-1, j-1)$ and so on.

Then we define

$$e(i,j) \quad := \quad \|v(l(.)) - (\bar{v}, \ldots, \bar{v})\|_D$$

for the grey level case, and

$$\begin{aligned} e(i,j) \quad := \quad & \|r(l(.)) - (\bar{r}, \ldots, \bar{r})\|_D \\ & + \|g(l(.)) - (\bar{g}, \ldots, \bar{g})\|_D \\ & + \left\|b(l(.)) - (\bar{b}, \ldots, \bar{b})\right\|_D \end{aligned}$$

for the RGB colour case, where $\bar{v}$, $\bar{r}$, $\bar{g}$, and $\bar{b}$ denote the mean values, i.e. $\bar{v} = \frac{v(l(1)) + \cdots + v(l(8))}{8}$ and so on.

In FIGURE 5, several different neighbourhoods and the corresponding values of $e(i,j)$ are shown. For the grey level case – which might be easily generalized to the colour case – we might observe the following:

OBSERVATION 1. $e(i,j)$ is zero in a completely homogeneous area.

All eight entries of the neighbourhood being equal means $v(l(1)) = v(l(2)) = \ldots = v(l(8)) = \bar{v}$, which implies $\|v(l(.)) - (\bar{v}, \ldots, \bar{v})\|_D = \|(0, \ldots, 0)\|_D = 0$.

OBSERVATION 2. $e(i,j)$ is relatively low when pixel values alternate between black and white, as is the case in half-tone, "chequerboard-like" areas.

Assume that there is an a such that the pixels at $(i-1,j)$, $(i,j-1)$, $(i+1,j)$ and $(i,j+1)$ — the "white" pixels of the chequerboard—have a value of $\bar{v}+a$, whereas the black pixels have a value of $\bar{v}-a$. Then $e(i,j) = \|(\bar{v}+a, \bar{v}-a, \bar{v}+a, \bar{v}-a, \ldots) - (\bar{v}, \ldots, \bar{v})\|_D$; since $\max X_b = \max(a, a-a, a-a+a, \ldots) = a$ and $\min X_b = \min(a, a-a, a-a+a, \ldots) = 0$, their sum is $e(i,j) = a$.

OBSERVATION 3. $e(i,j)$ has its maximum if the neighbourhood has a sequence of black pixels followed by another sequence of white pixels, as shown in the right-most example of FIGURE 5.

Assume that there is an a such that the first four pixels have a value of $\bar{v}-a$, the second consecutive four one of $\bar{v}+a$. Then $e(i,j) = \|(a,a,a,a,-a,-a,-a,-a)\|_D$; since $\max X_b = 4a$ and $\min X_b = 0$, $e(i,j) = 4a$.

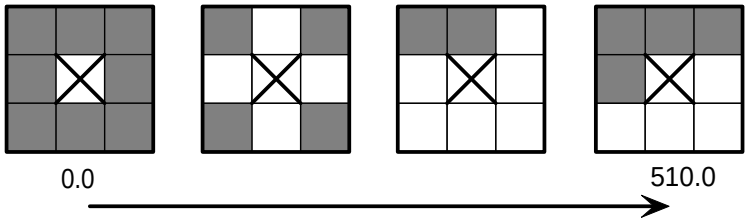


FIGURE 5: Several different neighbourhoods around the central pixel. $e(i,j)$ increases from left to right.

Out of these facts we can conclude that $e(i,j)$ indeed serves as an indicator to which degree the pixel is lying at or near an edge. Additionally, as tests in [BBK96] indicate, it is more robust w.r.t. noise than other conventional edge detectors.

4.2 Image enhancement

Image enhancement covers several different tasks such as removing or smoothing out noise while preserving or enhancing edges. In the past, the most common approach was to design a filter as sophisticated as possible which performed the enhancement on the complete image.

In recent years, another approach was investigated [CK95]. Instead of using one filter for the whole image, a filter bank with several filters of different characteristics is created. Then, a mechanism has to be developed which decides for each pixel which filter to apply, or, more generally, how to weigh each filter in the filter bank [CK95]. Fuzzy reasoning turned out to be a suitable filter-selecting mechanism; hence, we will follow that approach in this presentation.

4.3 Implementation of the enhancement view

The previous presentation of an image segmentation algorithm and its use to enhance images is independent from and not limited to our data/view model. Sure enough, however, I'd like to present this example as a final demonstration of the capabilities of the data/view model.

The first step in order to do this is to create or load an image and to "wrap it around" its boundaries; for this, I used a window view due to its internal usage of a cycle iterator.

In the next step, we have to form the neighbourhood of pixels around the central current pixel. Since we do not only want to calculate $e(i, j)$ out of this neighbourhood, but also apply a filter onto it, the complete 3×3 area is passed to the neighbour view, as shown in LISTING 11.

```
typedef boost::view::neighbour_view<WrappedImage, 9> View3x3;
View3x3::difference_type mask3x3( -stride-1, -stride, -stride+1,
                                      -1,         0,         +1,
                                  +stride-1, +stride, +stride+1 );
View3x3 view3x3( wrappedImage, mask3x3 );
```

LISTING 11: The "image enhancement view" operates on a 3×3 neighbourhood.

The implementation of the function computing the discrepancy norm of the neighbourhood is relatively straight-forward. The only point to watch is that, although its input values are all integer values in the range $\{0, \ldots, 255\}$, the computation has to be done with floating point precision to avoid rounding errors.

The calculated discrepancy norm and the variation are then taken as input for another function which determines the segmentation type of the area. This function wraps a very simple fuzzy system as described in [BBK96] with two input variables, one output variable, and five rules to decide between one of the four different types.

The final step is to use the computed segmentation type in order to decide which filter to take. The function object which combines all these steps is shown in LISTING 12.

What is left is to use this function object. Another view, namely a transform view, does take our ImageEnhancer function and applies it to the wrapped neighbour view. In code, this needs just another two lines:

```
typedef boost::view::transform_view<View3x3, ImageEnhancer> EnhancedView;
EnhancedView enhancedView( view3x3 );
```

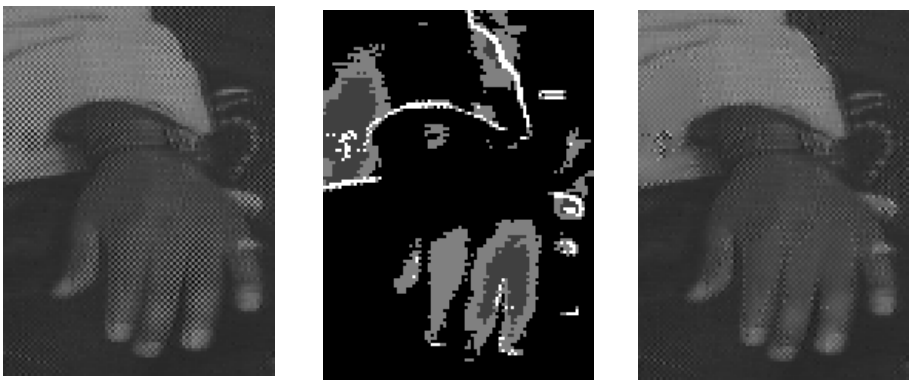


FIGURE 6: Original image, segmentation, and enhanced image.

```
struct ImageEnhancer:
  public std::unary_function< boost::tupple::n_fold_tuple<uchar,9>::type,
                              uchar >
{
  result_type operator()( const argument_type& u ) const
  {
    // Reorder neighbourhood elements as shown in FIGURE 4.
    boost::tupple::n_fold_tuple<uchar,8>::type arg(
      u.get3(), u.get0(), u.get1(), u.get2(),
      u.get5(), u.get8(), u.get7(), u.get6() );

    double dnorm = discrepancyNorm( arg );
    double var   = variance( arg );

    AreaType areaType = determineAreaType( dnorm, var );

    switch( areaType )
    {
    case Homogeneous: return filterPixel( u, ident );
      break;
    case Edge:        return filterPixel( u, sharpen );
      break;
    case Halftone:    return filterPixel( u, blur );
      break;
    case Picture:     return filterPixel( u, smooth );
      break;
    }
  }
};
```

LISTING 12: First the type of the neighbourhood is determined; then, an enhancing filter is selected accordingly.

The computations are relatively fast and take about 0.5 seconds for an 768×576 grey-level image. FIGURE 6 shows one example of a “noisy” image, its segmentation, and the computed enhanced image where the unwanted “chequerboard” artefacts are removed.

References

[Ara00] Kaoru Arakawa, *Fuzzy rule-based edge detection using multiscale edge images*, IEICE Trans. Fundamentals **E83-A** (2000), 291–300.

[BBK96] P. Bauer, U. Bodenhofer, and E. P. Klement, *A fuzzy algorithm for pixel classification based on the discrepancy norm*, Proc. FUZZ-IEEE’96, vol. III, 1996, pp. 2007–2012.

[CK95] YoungSik Choi and Raghu Krishnapuram, *Image enhancement base on fuzzy logic*, Proceedings of the 1995 International Conference on Image Processing (ICIP ’95), vol. 1, October 1995, pp. 167–170.

[Köt] Ullrich Köthe, *VIGRA - Vision with Generic Algorithms*, Cognitive Systems Group, University of Hamburg, Germany.

[Köt99] ———, *Reusable software in computer vision*, Handbook on Computer Vision and Applications (B. Jähne, H. Haußecker, and P. Geißler, eds.), vol. 3, Acadamic Press, 1999.

[Köt00] ———, *STL-Style Generic Programming with Images*, C++ Report Magazine **12** (2000), no. 1.

[Ric02] Roland Richter, *Tuples in C++ - a generative approach*, Tech. Report FLLL-TR-0214, Fuzzy Logic Lab Linz, 2002.

[Smi97] Steven W. Smith, *The scientist & engineer's guide to digital signal processing*, 1st ed., California Technical Pub., 1997.