Fuzzy Markup Language を用いたファジィシステムの開発

How to Develop Fuzzy Systems with Fuzzy Markup Language

○面﨑　祐一[1]，増山　直輝[1]，能島　裕介[1]，石渕　久生[2]
○Yuichi Omozaki[1], Naoki Masuyama[1], Yusuke Nojima[1], Hisao Ishibuchi[2]
[1]大阪府立大学　[2]南方科技大学
[1]Osaka Prefecture University [2]Southern University of Science and Technology

Abstract: The IEEE Std. 1855-2016 defines a W3C XML-based language called Fuzzy Markup Language (FML). Researchers can define fuzzy systems independent of the software. FML is composed of knowledge base and rule base. Knowledge base describes fuzzy sets. Rule base describes fuzzy rule sets. It can be defined flexibly, and in particular, the shape of membership functions can be defined by parameters easily. Besides, a program to calculate complex membership functions is provided in FML. This paper aims to introduce FML to researchers in the field of fuzzy systems and to provide a tool to define flexible membership functions. In addition, we demonstrate the effect of the tuned membership functions on the performance of fuzzy systems.

1. はじめに

近年，人工知能 (Artificial Intelligence: AI) の発展に伴い, AI の意思決定の判断根拠に対する不透明性が改めて問題視されている．特に，医療などの判断根拠の説明責任が求められる分野では，この不透明性が AI 導入の障壁となっている．そこで，意思決定のプロセスが，言語的に解釈可能なルール集合によって行われるファジィ識別器の透明性の高さに注目が集まっている [1].

ファジィ識別器はファジィシステムの一種であり，ファジィ集合で表現される言語ラベルを条件部として持つファジィ If-then ルールによって構成される識別器である．一般に，ファジィ集合はメンバシップ関数によって形式的に定義される．ファジィ識別器の識別境界はメンバシップ関数の形状に大きく依存しており，優れた識別器を設計するためには，言語ラベルを表現する適切なメンバシップ関数を定義する必要がある．

Fuzzy Markup Language (FML) [2] は IEEE Standard 1855-2016 で標準化された XML ベースの言語仕様である．現在，多くの研究者によってファジィシステムに関する研究が行われているが，一般に，研究者によって実装言語や実行環境が異なるため，プログラムの共有が困難である問題がある．そこで，実行環境に依存しない情報共有を可能にすることを目的として FML が標準化された. FML に基づくファジィシステムは知識ベースとルールベースのツリー構造で構成される．特に，知識ベースではファジィ集合が定義され，ファジィ集合を表現するメンバシップ関数はパラメータによって柔軟に定義される．また，FML に基づく XML ファイルの入出力を可能とする JFML という Java ベースのオープンソースライブラリが提供されている [3], [4].

本論文では，JFML を用いた実行環境に依存しないファジィシステム開発の普及を目的とし，JFML の導入を支援する GUI ツールを作成する．まず，JFML による開発を行うためには FML の仕様を熟知している必要がある．特に，メンバシップ関数の定義にはパラメータの設定が必要であり，各パラメータが形状に与える影響を理解する必要がある．そこで，スライダーや数値入力によってメンバシップ関数の形状をリアルタイムに描画する GUI ツールを作成し，JFML を初めて使用するユーザによるメンバシップ関数の定義を支援する．また，本ツールは SOFT-CR [5]または筆者らの GitHub リポジトリ [6]から利用することが可能である．

最後に，実際に本ツールを使用して設計したファジィ識別器の識別性能を調べる実験を行う．数値実験では，本ツールを用いて主観的に定義したメンバシップ関数を用いる場合と，均等に分割したファジィ集合を用いた場合とで，ファジィ識別器に与える影響を比較する．

2. ファジィ識別器の設計

n 次元 M クラスのパターンが m 個与えられたパターン識別問題に対して，ファジィ集合を条件部とする以下のファジィ If-then ルールを用いてファジィ識別器を設計する．1 つのパターンは $\mathbf{x} = (x_1, \ldots, x_n)$ のように表され，x_i は第 i 次元 ($i = 1, 2, \ldots, n$) における属性値を表す.

$$\text{Rule } R: \text{If } x_1 \text{ is } A_1 \text{ and } \ldots \text{ and } x_n \text{ is } A_n \text{ then Class } C \text{ with } CF, \quad (1)$$

ここで，$\mathbf{A} = (A_1, \ldots, A_n)$は条件部ファジィ集合，$C$ は結論部クラス，CF はルール重みを表す．また，各ルールの結論部クラスとルール重みは学習用データを用いて一意に決定される [7].

ファジィ集合はそれぞれメンバシップ関数で定義される．例えば，図 1 のような三角型ファジィ集合は以下の式(2)のように定義される．

$$\mu_{A_i}(x_i) = \max\left\{1 - \frac{|a - x_i|}{b}, 0\right\}, \quad (2)$$

$$a = (k-1)/(K-1), \quad b = 1/(K-1), \quad (3)$$

ここで，$\mu_{Ai}(\cdot)$は，ファジィ集合 A_i を定義するメンバシップ関数，K は分割数，k はその分割数におけるファジィ集合の番号を表す．同様にして，異なる形状のファジィ集合を定義するには，形状を表現するメンバシップ関数を定義する必要がある．

本論文では，条件部ファジィ集合の全組合せをルール集合とするファジィ識別器を設計する．また，未知パターンの推論には単一勝利ルール戦略を用いる．単一勝利ルール戦略では，入力パターンに対する適合度とルール重みの積が最大となるルールを勝者とし，そのルールの結論部クラスを推論結果とする．異なる結論部クラスを持つ複数のルールが等しく最大値をとるとき，その入力パターンを識別不能とする．

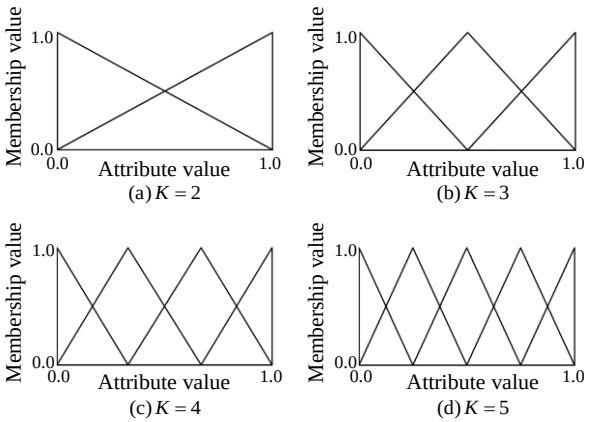

図 1　三角型ファジィ集合 (Triangular)

3. Fuzzy Markup Language

3.1. FML の仕様

FML は知識ベースとルールベースによって構成され，その構造が XML 形式で記述される．図 2 に FML のツリー構造の概略図を示す．

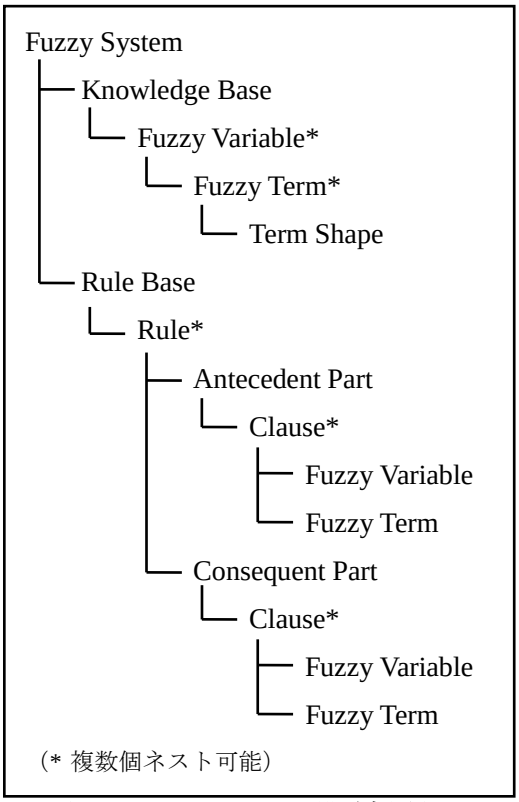

図 2　FML のツリー構造概略図

知識ベース (Knowledge Base) では，入力変数や出力変数をファジィ変数 (Fuzzy Variable) として定義し，各ファジィ変数に対して複数のファジィ集合 (Fuzzy Term) を定義することが可能である．さらに，ファジィ集合を表現するメンバシップ関数は，パラメータを XML 形式の属性値として与えることで定義される．ルールベース (Rule Base) では，複数のファジィ If-then ルール (Rule) が子要素としてネストされる．FML におけるファジィ If-then ルールは，前件部 (Antecedent Part) と後件部 (Consequent Part) で構成され，それぞれ複数の項 (Clause) を持つ．項は，知識ベースで定義したファジィ変数とファジィ集合の組合せで定義される．

3.2. FML で定義可能なメンバシップ関数の例

FML では，三角型，ガウス分布型，台形型などの様々な形状のメンバシップ関数が定義可能である．また，各種のメンバシップ関数はパラメータの値を変更することで柔軟にその形状を設定可能である [3]．以下に，FML で定義可能なメンバシップ関数の形状を例示し，図 3-5 にメンバシップ関数の定義例を示す．

- Triangular Shape（三角型）
- Left / Right Linear Shape（左/右上がり直線型）
- Trapezoid Shape（台形型）
- Rectangular Shape（区間型）
- Gaussian Shape（ガウス分布型）
- Singleton Shape（一点適合型）

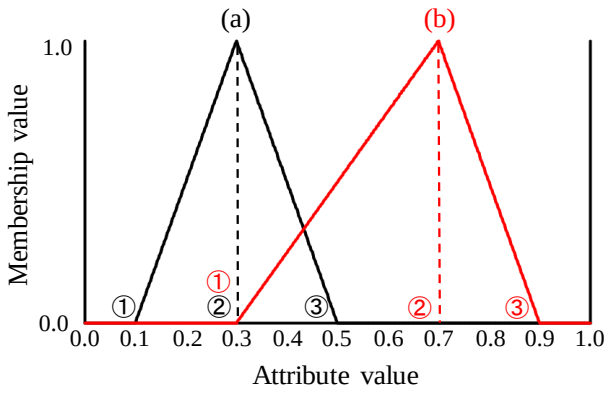

図 3　三角型メンバシップ関数の定義例

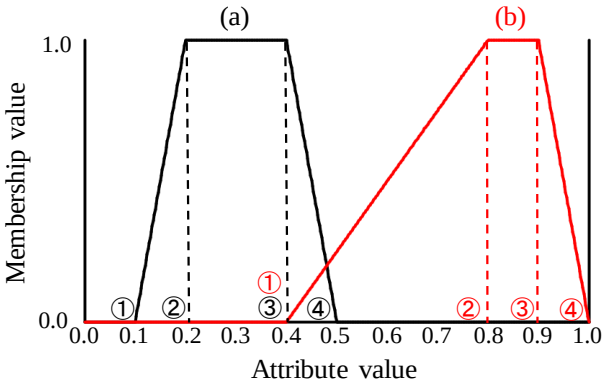

図 4　台形型メンバシップ関数の定義例

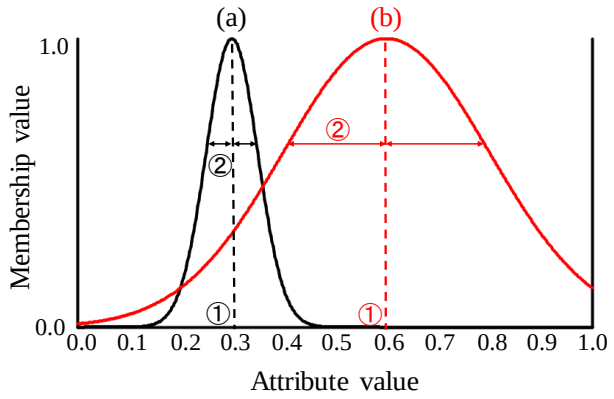

図 5　ガウス分布型メンバシップ関数の定義例

メンバシップ関数を定義するパラメータは，知識ベースにおいてファジィ集合を記述する XML 要素の属性 (param1 - param4) として設定される．例えば，図 3 の三角型メンバシップ関数はそれぞれ以下のパラメータで定義される．

(a) ① 0.1, ② 0.3, ③ 0.5
(b) ① 0.3, ② 0.7, ③ 0.9

同様に，図 4 の台形型メンバシップ関数はそれぞれ以下のパラメータで定義される．

(a) ① 0.1, ② 0.2, ③ 0.4, ④ 0.5
(b) ① 0.4, ② 0.8, ③ 0.9, ④ 1.0

また，ガウス分布型メンバシップ関数では，正分布を決定する平均と標準偏差の 2 つのパラメータで定義する．図 5 は以下のパラメータで定義したガウス分布型メンバシップ関数である．

(a) ① 0.3, ② 0.05
(b) ① 0.6, ② 0.20

3.3. JFML を用いたファジィシステムの開発

JFML [3]は，FML に基づいたファジィシステム開発を可能とする Java ベースのオープンソースライブラリである．さらに，前述のような様々な形状のメンバシップ関数の値を出力するプログラムがあらかじめ実装されている．JFML を使用することで，パラメータの設定のみで様々な形状のメンバシップ関数を扱うことが可能なため，複雑なメンバシップ関数を計算するプログラムを実装する必要がない．

また, JFML には XML ファイルの入出力を行うプログラムが提供されている．そのため，JFML を用いてファジィ識別器を設計することで，実行環境に依存しない情報共有が可能となる．

4. JFML による開発を支援する GUI ツール

JFML でメンバシップ関数を定義するためには，形状を定めるパラメータを設定する必要がある．そのため，ユーザはどのような形状のメンバシップ関数が定義可能であるかに加え，各パラメータによって形状がどのように変化するかについて理解しておく必要がある．そこで，本論文では JFML におけるメンバシップ関数の定義を支援する GUI ツールを作成する．本ツールではパラメータの設定を GUI で行うことが可能であり，さらに，メンバシップ関数の形状がリアルタイムに描画される．本ツールにより，JFML の初学者にも簡単にメンバシップ関数を定義することが可能となる．また，データセットの各次元におけるヒストグラムを表示する機能があり，データセットの分布を参考にしながらメンバシップ関数を主観的に定義することが可能である．本ツールの画面例を図 6 に示し，各種機能の説明をする．

A) JFML で定義可能なメンバシップ関数の種類をリストから選択する．種類を選択することでメンバシップ関数が更新され，選択した形状を確認することが可能である．

B) メンバシップ関数を定義するパラメータをスライダーの操作によって設定する．また，テキストボックスに値を直接入力して設定することも可能である．パラメータの値を設定することでメンバシップ関数がリアルタイムに更新される．

C) メンバシップ関数の形状を表示する．また，複数同時に表示することも可能である．

D) データセットの各属性におけるヒストグラムを表示する．また，クラスごとに色分けされ，積み上げ棒グラフで表示される．

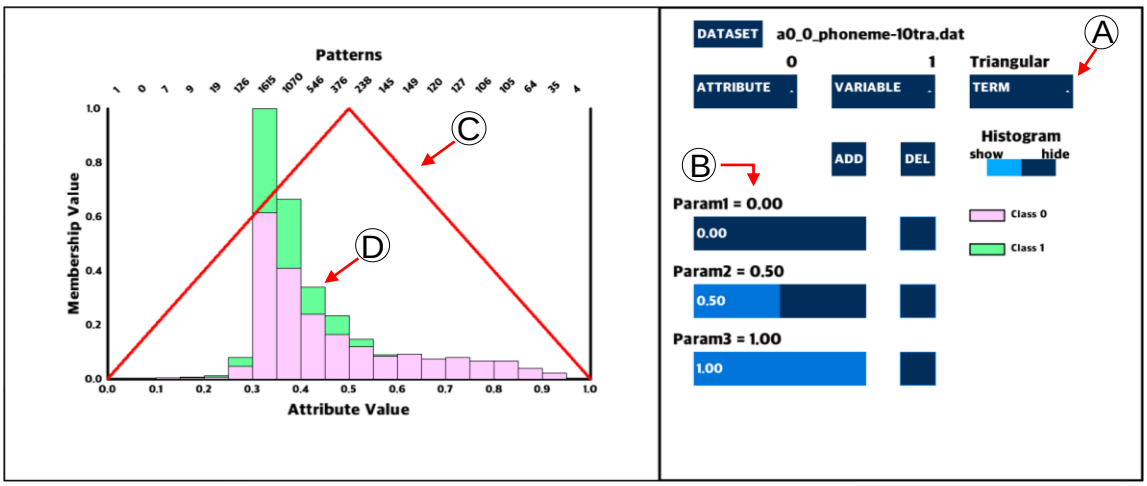

図 6　GUI 支援ツールの画面例

5. 数値実験

本章では，メンバシップ関数を主観的に定義することによるファジィ識別器の識別性能への影響を調査する．具体的には，主観的に定義したメンバシップ関数を用いる場合と，図 1, 7, 8 に示す均等に分割したファジィ集合を用いる場合とで，獲得される識別器の識別性能を比較する．ここで，両手法において獲得される識別器は共に，条件部ファジィ集合の全組合せをルール集合として設計される．また，均等に分割したファジィ集合を用いる場合，全ての入力次元を通して同一のファジィ集合を用いる．したがって，n 次元のデータに対し，K 分割のファジィ集合を用いるとき，識別器を構成するルールの総数は K^n となる．

比較実験として，均等に分割した三角型ファジィ集合（図 1）やガウス分布型ファジィ集合（図 7），台形型ファジィ集合（図 8）を用いたファジィ識別器について実験を行う．なお，これらのファジィ集合を定義するパラメータの詳細は，GUI ツールに同梱しているファイルを参照されたい．

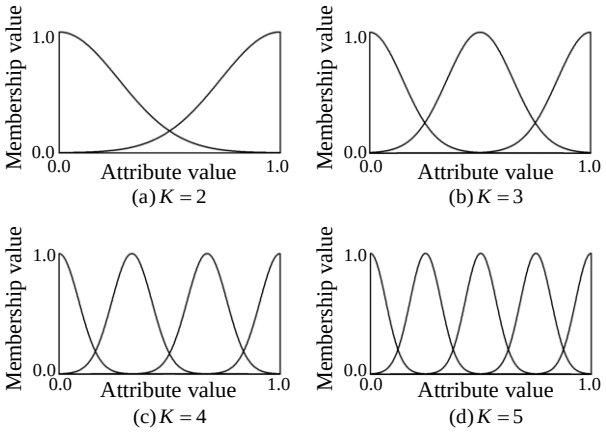

図 7　ガウス分布型ファジィ集合 (Gaussian)

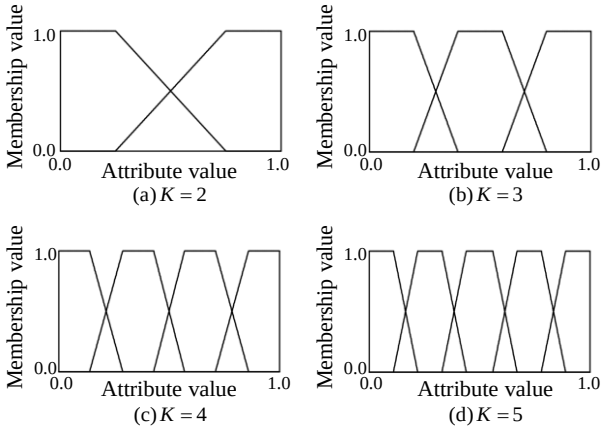

図 8　台形型ファジィ集合 (Trapezoid)

本数値実験では，実世界データとして UCI Machine Learning Repository [8]で提供されている中から，以下の表 1 に示す 3 種類のデータセットを使用した．

表 1　使用したデータセット

Dataset	Patterns	Attributes	Classes
Iris	135	4	3
Newthyroid	193	5	3
Phoneme	4,863	5	2

本論文では，GUI ツールを用いて，メンバシップ関数を以下のようにして定義する．まず，GUI ツールを用いて，入力次元ごとにヒストグラムを確認する．そして，各クラスの度数が最大となる区間でメンバシップ関数の値が大きくなるように定義する．この操作により，各メンバシップ関数が覆っているパターンのクラス比が偏り，各ルールの尤度が高くなり，識別精度が向上すると期待される．上記の操作に基づき定義したメンバシップ関数の形状とヒストグラムを図 9-11 に示す．また，メンバシップ関数を定義する各パラメータを表 2-4 に示す．

図 11 の 1 属性目のように複数のクラスで度数がピークとなる区間が等しい場合，前述の観点に基づいて定義することが困難となる問題がある．本実験ではこのような場合，筆者が主観的に定義したメンバシップ関数を使用した．

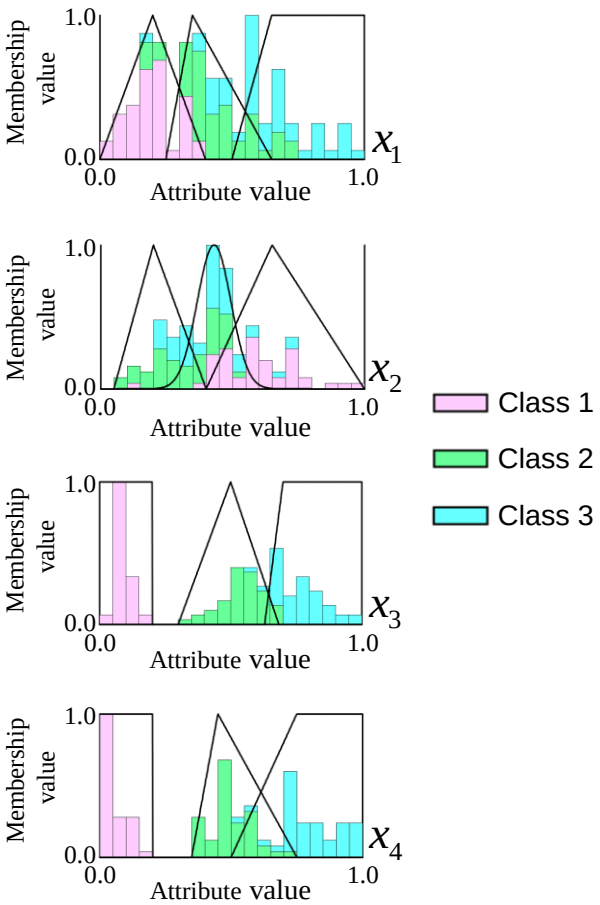

図 9　主観的に定義したメンバシップ関数とヒストグラム (Iris)

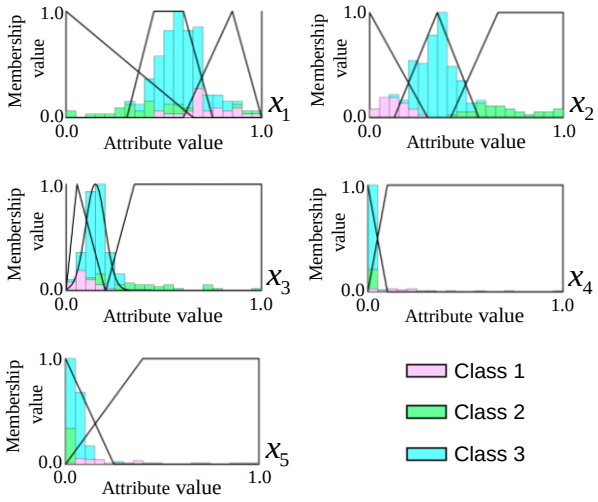

図 10　主観的に定義したメンバシップ関数とヒストグラム (Newthyroid)

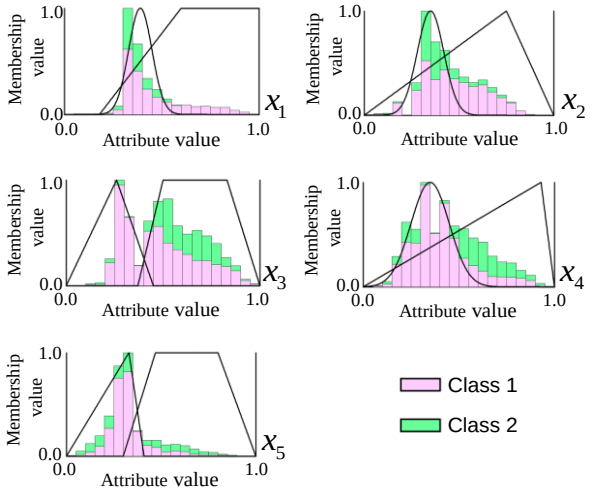

図 11　主観的に定義したメンバシップ関数とヒストグラム (Phoneme)

表 2　主観的に定めたパラメータ (Iris)

x_i		Parameters			
i	Shape	①	②	③	④
1	Trapezoid	0.50	0.65	1.00	1.00
	Triangular	0.00	0.20	0.40	-
	Triangular	0.25	0.35	0.65	-
2	Triangular	0.40	0.65	1.00	-
	Gaussian	0.43	0.06	-	-
	Triangular	0.05	0.20	0.40	-
3	Rectangular	0.00	0.20	-	-
	Triangular	0.30	0.50	0.68	-
	Trapezoid	0.63	0.70	1.00	1.00
4	Rectangular	0.00	0.20	-	-
	Trapezoid	0.50	0.75	1.00	1.00
	Triangular	0.35	0.45	0.75	-

表 3　主観的に定めたパラメータ (Newthyroid)

x_i		Parameters			
i	Shape	①	②	③	④
1	Trapezoid	0.31	0.45	0.60	0.75
	Triangular	0.60	0.85	1.00	-
	Triangular	0.00	0.00	0.65	-
2	Triangular	0.00	0.00	0.30	-
	Triangular	0.13	0.35	0.56	-
	Trapezoid	0.42	0.66	1.00	1.00
3	Trapezoid	0.20	0.35	1.00	1.00
	Gaussian	0.15	0.05	-	-
	Triangular	0.00	0.06	0.20	-
4	Triangular	0.00	0.00	0.10	-
	Trapezoid	0.00	0.10	1.00	1.00
5	Triangular	0.00	0.00	0.25	-
	Trapezoid	0.00	0.40	1.00	1.00

表 4　主観的に定めたパラメータ (Phoneme)

x_i		Parameters			
i	Shape	①	②	③	④
1	Gaussian	0.39	0.06	-	-
	Trapezoid	0.18	0.60	1.00	1.00
2	Gaussian	0.35	0.07	-	-
	Triangular	0.00	0.75	1.00	-
3	Triangular	0.00	0.26	0.45	-
	Trapezoid	0.37	0.50	0.83	1.00
4	Gaussian	0.35	0.10	-	-
	Triangular	0.00	0.93	1.00	-
5	Trapezoid	0.30	0.47	0.80	1.00
	Triangular	0.00	0.33	0.41	-

数値実験で得られた結果として，10-fold cross validation を 3 回繰り返した 30 回試行の誤識別率の平均を表 5-7 に示す．各データセットにおいて全識別器で最も良い結果を赤字で示し，GUI ツールを用いて設計した識別器と比較して悪い結果を青字で示す．

表 5-7 において，均等に分割したメンバシップ関数を用いたファジィ識別器の識別性能に注目すると，ほとんどの場合で分割数が大きいほど識別性能が向上していることが分かる．これは，分割数の増加に伴いルール数が増加することにより，識別境界の分解能が高くなるためだと考えられる．また，各データセットに対して最良の結果を示すメンバシップ関数の形状はそれぞれ異なっている．このことから，異なる形状のメンバシップ関数を用いることでファジィ識別器の識別性能に大きな影響を与えることが分かった．

GUI ツールを用いて設計したファジィ識別器は 3 分割および 2 分割のファジィ識別器と比較して，ほとんどの場合でよい結果を示している．また，主観的に設計したファジィ識別器では各属性に対して 2-3 個のファジィ集合を定義したため，ルール数が 3 分割のファジィ識別器と同数以下となる．このことから，データセットのヒストグラムに応じて主観的にメンバシップ関数を定義することは，少ないルール数で識別性能の高いファジィ識別器を設計するうえで有益であることが分かる．

表 5　Iris データに対する誤識別率

Classifier		Train	Test	# of
Shape	K	(%)	(%)	Rules
Triangular	5	3.33	4.44	625
	4	3.48	5.33	256
	3	6.22	6.44	81
	2	34.00	34.00	16
Gaussian	5	2.74	4.89	625
	4	7.73	12.67	256
	3	9.48	10.67	81
	2	26.96	27.33	16
Trapezoid	5	3.88	5.56	625
	4	4.64	7.78	256
	3	2.81	3.33	81
	2	14.99	15.33	16
GUI (図 9)		4.67	5.78	81

表 6　Newthyroid データに対する誤識別率

Classifier		Train	Test	# of
Shape	K	(%)	(%)	Rules
Triangular	5	6.41	7.42	3,125
	4	8.60	8.82	1,024
	3	14.09	14.07	243
	2	22.51	22.75	32
Gaussian	5	4.91	6.04	3,125
	4	4.63	6.81	1,024
	3	9.78	10.68	243
	2	16.06	16.42	32
Trapezoid	5	3.82	8.69	3,125
	4	5.06	7.75	1,024
	3	9.08	9.76	243
	2	18.59	18.74	32
GUI (図 10)		5.24	6.04	108

表 7　Phoneme データに対する誤識別率

Classifier		Train	Test	# of
Shape	K	(%)	(%)	Rules
Triangular	5	20.48	21.14	3,125
	4	21.74	22.11	1,024
	3	28.09	28.11	243
	2	29.35	29.34	32
Gaussian	5	17.57	18.89	3,125
	4	19.96	20.36	1,024
	3	28.31	28.61	243
	2	24.69	24.80	32
Trapezoid	5	16.60	17.75	3,125
	4	18.91	19.82	1,024
	3	25.53	25.97	243
	2	25.74	25.74	32
GUI (図 11)		24.43	24.38	32

6. おわりに

JFML は FML のツリー構造を実装した Java ベースのオープンソースライブラリであり，FML に基づく XML ファイルの入出力を可能にする．さらに，メンバシップ関数をパラメータの設定のみで扱うことができるため，複雑なメンバシップ関数を計算するプログラムを実装する必要がない．

本論文では，JFML を用いたファジィシステム開発を支援する GUI ツールを作成した．本ツールを用いることで，リアルタイムにメンバシップ関数の形状が確認可能であり，形状を定義するパラメータをユーザが理解することが可能となる．

また，本ツールを用いて主観的に定義したメンバシップ関数がファジィ識別器の識別性能に与える影響を調べる数値実験を行った．実験結果から，データセットのヒストグラムに基づきメンバシップ関数を定義することで，少ないルール数で高い識別性能を持つファジィ識別器の設計が可能であることが示唆される．しかしながら，図 11 の 1 次元目のようにデータセットの分布から主観的にメンバシップ関数を定義することが困難な場合がある．このような場合でユーザがメンバシップ関数を定義する際の指標として，データセットの分布から自動的に分割点を決定する手法で得られたメンバシップ関数を提示する機能の追加が今後の課題として挙げられる．

参考文献

[1] Y. Nojima and H. Ishibuchi, “Incorporation of user preference into multi-objective genetic fuzzy rule selection for pattern classification problems,” *Artificial Life and Robotics*, Vol. 14, pp. 418-421, 2009.

[2] *IEEE Standard for Fuzzy Markup Language*, Standard 1855-2016, 2016. (Available: https://standards.ieee.org/findstds/standard/1855-2016.html)

[3] Java Fuzzy Markup Language, (https://www.uco.es/JFML/)

[4] J. M. Soto-Hidalgo, Jose M. Alonso, G. Acampora, and J. Alcala-Fdez, “JFML: A Java library to design fuzzy logic systems according to the IEEE Std 1855-2016,” *IEEE Access*, Vol. 6, pp. 54952-54964, 2018.

[5] SOFT Computing Repository (http://soft-cr.org/)

[6] https://github.com/CI-labo-OPU/GUI_FMLtool.git

[7] H. Ishibuchi, T. Nakashima, and M. Nii, *Classification and Modeling with Linguistic Information Granules*, Springer-Verlag Berlin, Heidelberg, pp. 13-20, 2004.

[8] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

連絡先

増山　直輝
大阪府立大学大学院工学研究科
〒599-8531　大阪府堺市中区学園町 1-1
E-mail: masuyama@cs.osakafu-u.ac.jp