

УДК 004.827

doi: 10.15622/rcai.2025.036

СПОСОБ И ПРОГРАММНАЯ БИБЛИОТЕКА УЧЕТА СТРАТЕГИИ ОГРАНИЧЕНИЙ БАЗОВОГО ДИАПАЗОНА НЕЧЕТКИХ ЧИСЕЛ

М.М. Зернов (*zmmioml@yandex.ru*)^{A,B}

Н.А. Макеев (*n.a.makeenkov@yandex.ru*)^A

^A АО «Радиозавод», Пенза

^B Филиал Национального исследовательского университета «МЭИ»
в г. Смоленске, Смоленск

В работе предлагается способ учета стратегии ограничений базового диапазона нечетких чисел на основе функций высшего порядка, позволяющий упорядочить интерпретацию нечеткого числа с учетом ограничений. Способ нашел применение в разрабатываемой программной библиотеке нечетких вычислений на языке C++. Структура программных классов библиотеки построена с применением шаблонов проектирования «стратегия», CRTP, «посетитель». Использован дополнительный прием отсечения элементов, выходящих за пределы носителя, при расчете функции принадлежности для упорядоченного массива входных чисел. Проведена экспериментальная оценка быстродействия, подтвердившая эффективность реализации способа.

Ключевые слова: функции высшего порядка, нечеткая математика, нечеткие числа.

Введение

Область определения нечеткого числа в ситуациях, когда оно определено не на всей числовой оси, а на ограниченном базовом диапазоне (интервале), редко учитывается при проведении арифметических операций. Можно задать ограничение на множество возможных пар элементов базового множества при определении операций по принципу расширения Заде, т.е. использовать совместное распределение возможностей [Fuller et al., 2004], [Carlsson et al., 2004]. Однако, гораздо эффективнее использовать совместные распределения возможностей для других целей, например, уменьшения размытия результата нечетких операций при итерацион-

ных вычислениях [Федулов, 2006]. Всевозможные способы выполнения нечетких арифметических операций на основе L-R чисел, как и большинство способов на основе α -уровневых множеств не предусматривают учета границ базового диапазона. Фактически, единственной областью, в которой такой контроль осуществляется, является нечеткий логический вывод, в котором, в принципе, границы области определения лингвистических переменных используются лишь при вычислении значений функций принадлежности. При этом, по сути, используется стратегия отсечения части нечеткой переменной (заданной нечетким числом) не попавшей в область определения. В работе [Зернов, 2008] было предложено преобразование интерпретации результата нечетких арифметических операций на ограниченном базовом диапазоне, использующее другую стратегию – агрегации посредством S-нормы. Смысл преобразования заключается в том, что, чем более значимая часть результата нечеткой арифметической операции выходит за границу базового диапазона, тем выше принадлежность соответствующей границы диапазона результату операции. Иначе говоря, утверждения “число X меньше a ” и “число X больше b ” лишены физического смысла и должны трактоваться как “Число X равно a ” и “Число X равно b ” соответственно.

Однако его применение можно расширить и на случай любого нечеткого числа, в т.ч. заданного параметрически. Пусть имеем нечеткое число с функцией принадлежности μ , заданное на интервале действительных чисел $[a, b]$. Тогда скорректированное значение принадлежности μ' можно представить как:

$$(1.1)$$

Естественно, такое преобразование является лишь одним из вариантов интерпретации нечеткого числа в границах базового диапазона. Помимо этого, могут использоваться другие S-нормы, могут вовсе не накладываться ограничения. Наконец, может быть применено и простое отсечение:

$$(1.2)$$

Далее в работе будет предложен формализм способа учета стратегии ограничений и представлена программная библиотека на языке C++, в которой реализован предложенный способ.

1. Обзор библиотек нечетких вычислений

В настоящее время имеется большое количество программных библиотек в той или иной степени, ориентирующихся на флагманское решение – пакет Fuzzy Logic Toolbox для работы с системами нечеткого логического вывода [Alcal'a-Fdez et al., 2016]. Основной функционал построен вокруг трех областей – построения систем нечеткого логического вывода, нечеткой кластеризации, обучения систем нечеткого логического вывода на основе гибридных ANFIS-моделей [Jang, 1993]. Позднее к нему были добавлены возможности построения систем нечеткого логического вывода на основе нечетких множеств 2 типа [Liang et al., 2000]. Примечательно, что Fuzzy Logic Toolbox также предоставляет функцию *fuzarith*, позволяющую выполнять арифметические операции над нечеткими числами, заданными на одном и том же базовом диапазоне, с которым соотносится и результат. Вычисления осуществляются методом α -уровней [Dong et al., 1985]. Операнды задаются как массивы нечетких значений, соотносимых с одним универсальным множеством. К нему же относится и результат. Таким образом, реализуется стратегия ограничения результата операции путем отсеечения. Однако, работа с нечеткими числами является скорее побочной функцией и реализована не самым удобным образом, особенно с учетом не самых плохих возможностей объектно-ориентированного программирования в Matlab. В любом случае, решение является проприетарным и нишевым.

Другой заметной программной библиотекой, ориентированной на нечеткие вычисления и предоставляющей операции над нечеткими числами является scikit-fuzzy – пакет (набор библиотек) для языка Python, совместимая с другим известным пакетом scikit-learn [Warner et al., 2019], [Zhang et al., 2024]. Здесь вычисления выполняются без ограничений. При этом используется два метода – на основе принципа расширения Заде [Zadeh, 1975] и метод на основе α -уровней. В последнем случае, в отличие от реализации Fuzzy Logic Toolbox, операнды могут иметь разные базовые диапазоны, а базовый диапазон результата вычисляется на их основе.

Несмотря на развитую объектную модель в отношении систем нечеткого логического вывода, нечеткие числа, как таковые, отдельными классами не представлены.

По результату анализа распространенных программных библиотек для реализации систем нечеткого логического вывода и нечетких вычислений, авторами было принято решение о разработке собственной объектно-ориентированной библиотеки на языке C++ 17, позволяющей варьировать стратегии ограничения базового диапазона для разнообразных функций принадлежности нечетких чисел. Помимо этого библиотека должна позволять осуществлять арифметические операции над нечеткими числами с применением различных методов. Сами операции должны, по возможности, позволять естественную запись на основе математических операторов над объектами, представляющими нечеткие числа.

2. Способ учета стратегии ограничений базового диапазона нечетких чисел

Пусть имеется множество стратегий ограничений базового диапазона $S = \{s_0, s_1, \dots, s_k\}$, состоящее минимум из двух элементов – тривиальной стратегии s_0 “без ограничений” и хотя бы одной “ограничивающей”, например на основе агрегации через *max* выпадающей части нечеткого числа. Каждая стратегия s представлена функционалом (функцией высшего порядка) $F_s(\mu, [a, b])$ от функции принадлежности и базового диапазона, возвращающим скорректированное значение функции принадлежности. Для стратегии “без ограничений” функционал тривиален:

Для ограничивающей стратегии на основе агрегации через *max* – определяется в соответствии с выражением (1.1). Для ограничивающей стратегии на основе отсечения – в соответствии с выражением (1.2).

Таким образом, нечеткое число с функцией принадлежности μ характеризуется тройкой:

При этом, разумно наложить ограничение – вся числовая ось в качестве базового диапазона может быть использована только совместно с тривиальной стратегией:

Иначе говоря, бессмысленно использовать “ограничивающую” стратегию, не зная базового диапазона, равно как и устанавливать базовый диапазон, не задавая нетривиальной стратегии.

Таким образом, способ учета стратегии ограничений базового диапазона нечетких чисел на основе функций высшего порядка выражается в специфике выполнения следующих действий.

1. Создание/инициализация нечетких чисел.

Вначале нужно определить нетривиальную стратегию ограничений, используемую для создания нового “ограниченного” числа. Затем при создании нового нечеткого числа, ответить на вопрос известно ли ограничение на базовый диапазон для него в текущем моменте/месте производимых вычислений.

Если ответ положительный – создаем нечеткое число как совокупность функции принадлежности, “ограничивающей” стратегии и нетривиального базового диапазона.

Иначе – как совокупность функции принадлежности, тривиальной стратегии “без ограничений” и тривиального базового диапазона.

2. Наложение ограничений на уже имеющееся нечеткое число.

Может производиться как над неограниченным числом, так и числом с нетривиальным базовым диапазоном. В обоих случаях следует создать новое нечеткое число в виде совокупности функции принадлежности исходного числа, выбранной в п.1 ограничивающей стратегии и задаваемого диапазона.

3. Вычисление значения функции принадлежности ограниченного числа в заданной точке.

Вычисление делегируется функции высшего порядка, представляющей стратегию. При этом, наивная реализация такой стратегии, как например, “агрегация через *max*” может быть вычислительно затратной. Вместо прямого использования выражения (1.1), предлагается в случае вычисления на границах диапазона сопоставлять взаимное расположение границы и супремума нечеткого числа.

Итак, пусть имеется функция $f(x)$, возвращающая пару

из собственно супремума функции принадлежности (который может быть вычислен, например, из ее параметров) на всей числовой оси и соответствующего ему интервала α -уровня. Тогда выражение 1.1 может быть переписано в виде:

Строго говоря, $\alpha_{sup} = 1$ в силу условия нормированности нечеткого числа, а сам соответствующий интервал α -уровня представляет собой моду (для унимодальных чисел) или интервал модальных значений (для толерантных чисел). Однако, это условие предлагается ослабить и применять предложенное преобразование для всех нечетких множеств, заданных выпуклыми функциями принадлежности.

Отдельно стоит рассмотреть вопрос применения ограничений к нечеткому числу, заданному гистограммой. В этом случае (при выполнении условия выпуклости функции принадлежности), предлагается использовать его аппроксимацию на основе множеств α -уровня.

Если же условие выпуклости функции принадлежности не выполняется, то речь идет уже о просто нечетком множестве, определенном на множестве действительных/целых чисел и для него в принципе целесообразно определить как набор поддерживаемых операций, так и способов их реализации. Предложенный способ и его программная реализация применимы к нечетким множествам с выпуклой функцией принадлежности.

3. Объектная модель библиотеки FuzzyLib++

В данном разделе представлена объектная модель разрабатываемой библиотеки нечетких вычислений на языке C++ – FuzzyLib++. В настоящий момент реализована только часть планируемого функционала библиотеки, а именно, управление стратегией ограничения нечетких чисел.

На рис. 1. представлена диаграмма классов библиотеки. Объектная модель использует три распространенных шаблона проектирования – “стратегия”, CRTP (Curiously Recursive Template Pattern) и “посетитель” [Александреску, 2008].

Полиморфизм реализован статически на основе CRTP-шаблона, в котором класс нечеткого множества *FuzzySet* выступает в качестве базы наследования, предоставляя общие методы нечетких множеств – получения типа множества, расчета значения функции принадлежности. Последний подразумевает несколько реализаций – как для передаваемого буфера результатов, так и вновь формируемого, а также для упорядоченных и неупорядоченных входных значений.

Промежуточным звеном иерархии наследования является класс нечеткого числа *FuzzyNumber*, использующий в качестве носителя – числа с плавающей точкой. Нечеткое число является точкой сборки стратегий – функции принадлежности *MF* и шаблонной стратегии ограничения *TLimiter*, смешивая их с функционалом нечеткого множества. Нечеткое число добавляет методы получения интервала носителя, модального интервала/интервала супремума, набора (вектора) α -уровней.

В соответствии с шаблоном CRTP, базовый класс в качестве шаблонного аргумента принимает класс наследника. Таким образом, класс наследника наследует неявно заданному абстрактному интерфейсу. В нашем случае, наследование от *FuzzySet* подразумевает следование интерфейсу *CFuzzySet* (конкретный тип нечеткого множества), определяющему наличие реализаций методов расчета значений функции принадлежности в точке и на массиве входных значений с размещением в готовом буфере. Наследование от шаблонного класса нечеткого числа более сложно. Наследниками являются стратегии ограничений, примененные к функциям принадлежности. Помимо методов конкретного нечеткого множества, они должны предоставлять методы функции принадлежности – частные реализации получения всех интересующих интервалов. Для того, чтобы ме-

тоды стратегий ограничения реализовывались с учетом результатов конкретных функций принадлежности – они наследуются от своих шаблонных параметров.

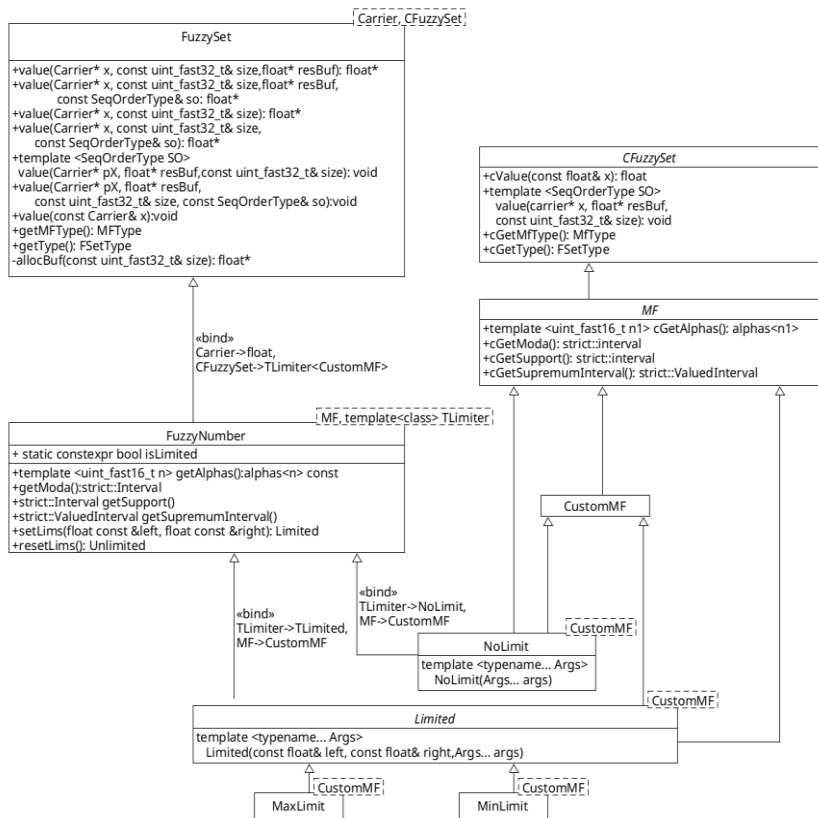


Рис. 1. Общая диаграмма классов библиотеки FuzzyLib++

На рис. 2 представлена диаграмма классов для результата подстановки (инстанцирования) в используемые шаблоны конкретного класса 2-сторонней гауссовой функции принадлежности. Стратегия отсечения в текущей версии не реализована, но подразумевается возможность ее внедрения.

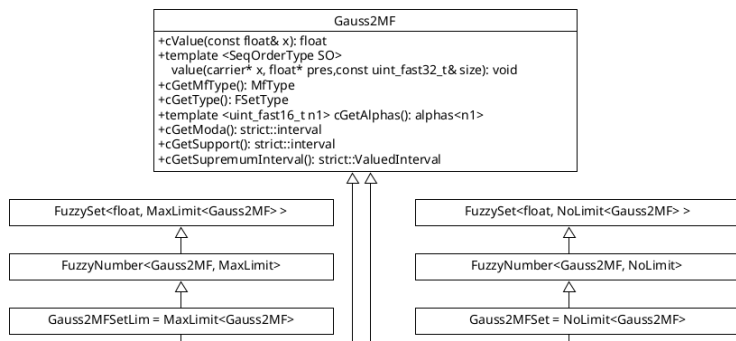


Рис. 2. Диаграмма классов, генерируемых для 2-сторонней гауссовой функции принадлежности

Слабой стороной шаблона проектирования CRTP является то, что для каждого конкретного наследника (например, неограниченного и ограниченного в соответствии со стратегией агрегации по максимуму нечеткого числа с 2-сторонней гауссовой функцией принадлежности), при подстановке строится своя иерархия наследования. Статический полиморфизм обеспечивает большее быстродействие, что особенно критично для встраиваемых систем по сравнению с динамическим (виртуальными методами, как способом реализации абстрактных интерфейсов). С другой стороны, к наследникам уже не обратиться через указатель на абстрактный интерфейс.

В результате нужно или явно использовать каждый такой конкретный класс нечеткого числа или создавать сигма-тип объединяющий наследников. Для последнего необходимо явное указание входящих в него вариантов и определение основных вызовов с использованием шаблона “посетитель”.

Рассмотренная объектная модель реализована в рамках стандарта C++17, предоставляющего средства как для реализации сигма-типов (шаблонный класс `std::variant`), так и для их посещения (шаблонная функция времени компиляции `std::visit`).

4. Результаты оценки производительности вычисления значений функций принадлежности в библиотеке FuzzyLib++

Тестирование проводилось на вычислительной машине с процессором Intel(R) Xeon(R) CPU E5-2680 v4 с базовой частотой 2.40 ГГц и максимальной 3.30 ГГц под управлением операционной системы Fedora Linux 41 Server с оперативной памятью 112 Гбайт DDR4 работающей на эффективной частоте 2400 МГц. В качестве компилятора был использован

гсс 13.3. Использован уровень оптимизации компилятора «-O2» и флаг оптимизации под архитектуру процессора «-march=native». Использовалась система сборки *cmake*.

Задавалось различное количество (101, 1001, 10001) входных действительных значений, которые в виде массива подавались на вход метода вычисления значения функции принадлежности нечеткого числа. С целью подсчета чистой производительности без учета аллокации буфера результата в куче, использовался метод класса нечеткого множества, размещающий рассчитанные значения в буфере результата, переданном в виде указателя.

В табл. 1 представлено сравнение времени вычисления нечеткого числа с гауссовой функцией принадлежности с параметрами $M = 5$, $\sigma = 1,67$ для 10001 значения на интервале $[-3,01354, 13,01345]$ без ограничения и с ограничением базового диапазона интервалом $[5,5, 20]$.

Таблица 1

Время вычисления 10001 значения функции принадлежности нечеткого числа гауссового типа

Количество значений 10001	Попадание в носитель, отн.ед.	t возр., нс	t неуп., нс
Неограниченное число	1	93182	92839
Ограниченное число	0,468	64409	140995
Отн. разница, %	-53,2	-30,9	51,9

Использовались два варианта метода вычисления значения функции принадлежности – с учетом порядка входных значений по возрастанию и без. Порядок имеет смысл только для реализации гауссового числа с ограничениями – отсекаются части входного массива, не попадающие в ограничения и носитель нечеткой функции принадлежности. Неограниченная реализация такой оптимизации пока не имеет. Доля входных значений, попавших в носитель нечеткого числа показана в первом столбце таблицы. В каждом случае указано среднее время по результатам 50 измерений.

Как видно, в общем, неупорядоченном, случае, время вычисления значений функции принадлежности в ограниченном варианте почти на 52% превышает время неограниченного варианта. Тем не менее, расчет результата с размещением в готовом буфере, даже с учетом накладных расходов на реализацию стратегии агрегации по максимуму, не превышает в среднем 0,2 мс. Такой результат позволяет надеяться на достаточное быстроедействие в режиме реального времени и для встраиваемых систем. Последнее утверждение, конечно, в дальнейшем, предстоит проверить экс-

периментально – скомпилировав библиотеку для распространенных архитектур микроконтроллеров и проведя испытания на соответствующих стендах. Заметим, что оптимизация по пересечению с носителем позволяет частично компенсировать накладные расходы на применение стратегии ограничения.

Аналогичное сравнение проведено и для меньшего количества входных значений – 1001, результаты которого представлены в табл. 2.

Таблица 2

Время вычисления 1001 значения функции принадлежности нечеткого числа гауссовой типа

Количество значений 1001	Попадание в носитель, отн.ед.	t возр., нс	t неуп., нс
Неограниченное число	1	9525	9428
Ограниченное число	0,468	7040	15854
Отн. разница, %	-53,2	-26,1	68,2

Как видно, с уменьшением числа рассчитываемых значений, при той же их пропорции между ограниченным и неограниченным случаем, доля накладных расходов на расчет ограничений растет. Тем не менее, общее время расчета не превышает 16 мкс.

В табл. 3. Представлены результаты измерения времени расчета 10001 значения функций принадлежности нечетких чисел с гауссовой, двусторонней гауссовой, треугольной функциями принадлежности, а также, функции принадлежности на основе 101 α -уровневого интервала (получены из двусторонней гауссовой функции принадлежности). Для последней, результат рассчитывается путем линейной интерполяции по границам двух ближайших интервалов α -уровня.

Ограниченные версии чисел рассчитывались с различной пропорцией попадания входных значений в носитель ограниченного числа. Для неограниченных чисел, практически, все значения попадали в носитель.

Таблица 3

Время вычисления 10001 значения функции принадлежности нечетких чисел различных типов

Тип ФП	Попад. в носит. огр., отн.ед.	Отн. разн, %	t неогр., нс	t огр. неуп., нс	Отн. разн, %	t огр. возр., нс	Отн. разн, %
GaussMF	0,468	-53,2	93182	140995	51,3	64409	-30,9
Gauss2MF	0,595	-40,5	94230	152769	62,1	70114	-25,6
TriMF	0,250	-75,0	491517	405236	-17,6	146047	-70,3
Alphas101	0,595	-40,5	2375444	866877	-63,5	1634639	-31,2

Как видно, наиболее сложной в смысле времени вычислений является функция принадлежности на основе α -уровней. Но даже для нее время расчета 10001 значения составляет порядка единиц мс.

Для ограниченных чисел на основе гауссовой и 2-сторонней гауссовой функций принадлежности в общем, неупорядоченном, случае, время вычисления значений функции принадлежности растет.

Упорядоченный вариант позволяет компенсировать часть накладных расходов, пусть и не в той же пропорции, в которой уменьшается доля входных значений, попадающих в носитель.

Для треугольной функции принадлежности и функции принадлежности на основе α -уровней, в силу их интервального характера, даже неупорядоченный вариант вычислений приводит к уменьшению времени вычислений при «выпадании» элементов носителя. Причем, для последней, «улучшенный» вариант, учитывающий порядок значений делает это в среднем хуже. Такие результаты говорят о необходимости дальнейшей оптимизации как процедуры отсечения «лишних значений» так и процедуры интерполяции.

Заключение

Представленная программная библиотека реализует пока только простейший функционал. Тем не менее, в ней уже заложены принципы, отличающие ее от других библиотек нечетких вычислений – представление нечеткого числа в виде объекта, возможность гибкого применения стратегий ограничения нечетких чисел, возможность восстановления исходной формы ограниченного нечеткого числа. Таким образом, реализуется предложенный способ учета стратегии ограничений базового диапазона нечетких чисел на основе функций высшего порядка.

В дальнейшем, планируется добавление в библиотеку нечетких арифметических и логических операций с возможностью задействовать различные комплекты реализации данных операций.

Предполагается публикация библиотеки под свободной лицензией.

Список литературы

- [Александреску, 2008] Александреску А. Современное проектирование на C++: пер. с англ. – М.: Издательский дом «Вильямс», 2008. – 336 с.
- [Зернов, 2008] Зернов М.М. Арифметические вычисления над нечеткими числами на ограниченном базовом диапазоне // Тез. докл. IX Междунар. конф. «Системы компьютерной математики и их приложения» (СКМП-2008). – Смоленск, 2008. – С. 42-44.
- [Федулов, 2006] Федулов А.С. Вид взаимодействия нечетких чисел, ограничивающий возрастание неопределенности при выполнении операций нечеткой арифметики // Вестник МЭИ. – 2006. – № 1. – С. 101-110.

- [Alcal'a-Fdez et al., 2016] Alcal'a-Fdez J., Alonso J.M. A Survey of Fuzzy Systems Software: Taxonomy, Current Research Trends, and Prospects // IEEE Transactions on Fuzzy Systems. Feb. 2016 . – Vol. 24, No. 1 . – P. 40 -56 (статья в журнале на англ. языке).
- [Carlsson et al., 2004] Carlsson C., Fuller R., Majlender P. Addition of completely correlated fuzzy numbers // FUZZ – IEEE 2004. IEEE Int. Conf. on Fuzzy Systems. Budapest, Hungary, 2004.
- [Dong et al., 1985] Dong W., Shah H., Wong F., Fuzzy computations in risk and decision analysis // Civ Eng Syst. – 1985. – 2. – P. 201-208.
- [Fuller et al., 2004] Fuller R., Majlender P. On interactive fuzzy numbers // Fuzzy Sets and Systems. – 2004. – Vol. 143. – P. 355-369.
- [Jang, 1993] Jang J.-S. ANFIS: adaptive-network-based fuzzy inference system // IEEE Transactions on Systems, Man, and Cybernetics. – 1993. – Vol. 23, No. 3. – P. 665-685.
- [Liang et al., 2000] Liang, Q., & Mendel, J. M. Interval type-2 fuzzy logic systems: theory and design // IEEE Transactions on Fuzzy systems. – 2000. – Vol. 8, No. 5. – P. 535-550.
- [Warner et al., 2019] Warner J., Sexauer J., scikit-fuzzy, twmeggs, alexsavio, Unnikrishnan A., Castelaõ, G., Pontes, F.A., Uelwer, T., pd2f, laurazh, Batista, F., alexbuy, Broeck, W.V. den, Song, W., Badger, T.G., Pe´rez, R.A.M., 7002 Power, J.F., Mishra, H., Trullols, G.O., Ho¨rteborn, A., 99991, 2019. JDWarner/scikit-fuzzy: Scikit-Fuzzy version 0.4.2. – <https://doi.org/10.5281/zenodo.3541386>.
- [Zadeh, 1975] Zadeh L.A. Concept of a linguistic variable and its application to approximate reasoning // I, II, III, Information Sciences. – 8(1975) 199-249, 301-357; 9(1975). 43-80.
- [Zhang et al., 2024] Zhang D., Chen T. Scikit-ANFIS: A Scikit-Learn Compatible Python Implementation for Adaptive Neuro-Fuzzy Inference System // International Journal of Fuzzy Systems. – 2024. – 26. – 10.1007/s40815-024-01697-0.