МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

РЕЗНІК РОМАН ЮРІЙОВИЧ

Допускається до захисту:
завідувач кафедри
інформаційних технологій,
д. т. н., доцент
_____ Т. В. Нескородєва
«_____» _____ 2022р.

ІДЕНТИФІКАЦІЯ БАГАТОФАКТОРНИХ ЗАЛЕЖНОСТЕЙ НА ОСНОВІ НЕЧІТКИХ БАЗ ЗНАНЬ З СУПЕРЕЧЛИВИМИ ПРАВИЛАМИ

Спеціальність 122 «Комп'ютерні науки»

Кваліфікаційна (магістерська) робота

Науковий керівник:
Штовба С.Д., професор кафедри
інформаційних технологій,
д-р. техн. наук, професор
_____
(підпис)

Оцінка: _____/_____/_____
(бали за шкалою ЄКТС/за національною шкалою)
Голова ЕК: _____
(підпис)

Вінниця – 2022

АНОТАЦІЯ

Резнік Р.Ю. Ідентифікація багатофакторних залежностей на основі нечітких баз знань з суперечливими правилами**.** Спеціальність 122 «Комп'ютерні науки». Донецький національний університет імені Василя Стуса, Вінниця, 2022.

У кваліфікаційні роботі пропонується новий підхід в синтезі та відборі правил нечітких баз знань, а саме застосування суперечливих правил.

В результаті, встановлено, що застосування таких суперечливих правил підвищує «роздільну здатність» опису залежності, яку моделює база знань. Відповідно, точність нечіткої бази знань за рахунок використання суперечливих правил - зростає.

Ключові слова: нечітка логіка, Мамдані, нечітка база знань, відбір правил.

Табл. 16. Рис. 43. Бібліограф.:50 найм.

ABSTRACT

Reznik R.Y. Identification of multifactor dependence with usage of fuzzy knowledge base with conflicting rules. Specialty 122 “Computer Science”. Vasyl’ Stus Donetsk National University, Vinnitsa, 2022.

A new approach in the synthesis and selection of rules of fuzzy knowledge bases is proposed in the qualification work, namely the application of conflicting rules.

As a result, it was established that the application of such conflicting rules increases the "resolution" of the description of the dependence modeled by the knowledge base. Accordingly, the accuracy of the fuzzy knowledge base due to the use of contradictory rules increases.

Keywords: fuzzy logic, Mamdani, fuzzy knowledge base, rule selection.

Tabl. 16. Fig. 43. Bibliography: 50 items.

ЗМІСТ

ВСТУП 5
РОЗДІЛ 1 ОГЛЯД СТАНУ ПИТАННЯ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ 8
1.1 Основні положення теорії нечітких множин 8
1.2 Нечітка база знань. База знань Мамдані 11
1.3 Огляд методів відбору правил нечіткої бази знань 14
1.4 Синтез правил нечітких баз знань, алгоритм Ванга-Менделя 16
1.5 Ідея дослідження та деталізація завдань 19
Висновки до розділу 1 20
РОЗДІЛ 2 МОДЕЛІ ТА АЛГОРИТМИ ВИРІШЕННЯ ЗАДАЧІ ВІДБОРУ ПРАВИЛ НЕЧІТКИХ БАЗ ЗНАНЬ 22
2.1 Модифікація алгоритму Ванга-Менделя – «сума голосів» 22
2.2 Модифікація алгоритму Ванга-Менделя – «головний конкурент» 24
2.3 Синтез суперечливих нечітких баз знань з підмножини суперечливих, метод найближчого сусіда 28
2.4 Синтез несуперечливих нечітких баз знань з підмножини суперечливих 30
Висновки до розділу 2 32
РОЗДІЛ 3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ 33
3.1 Нечітка ідентифікація в системі Matlab 33
3.2 Архітектура системи 41
3.3 Модуль реалізації адитивних та субтрактивних алгоритмів відбору правил нечітких баз знань 42
3.4 Модуль реалізації класичного алгоритму Ванга-Менделя та запропонованих модифікацій 44
3.5 Додаткові модулі 46
Висновки до розділу 3 46
РОЗДІЛ 4 ЕКСПЕРЕМЕНТАЛЬНІ ДОСЛІДЖЕННЯ 47
4.1 Методика проведення досліджень 47
4.2 Формування баз знань 48
4.3 Генерація тестової вибірки 54

4.4 Синтез баз знань з підмножини суперечливих та несуперечливих правил 57
4.5 Синтез несуперечливих баз знань з підмножини суперечливих правил 65
4.6 Класичний алгоритм Ванга-Менделя для синтезу нечітких правил баз знань та запропоновані варіанти покращення алгоритму 67
4.7 Якість досліджених методів 69
Висновки до розділу 4 70
ВИСНОВКИ 71
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ТА ЛІТЕРАТУРИ 72
ДОДАТКИ 77

ВСТУП

Актуальність теми дослідження

На сьогоднішній день все частіше зустрічаються системи які розроблені на основі нечітких баз знань. Нечіткі бази знань, що являють собою сукупність висловлювань “якщо –то”, є ефективним засобом моделювання в багатьох задачах кібернетики:

- управління технологічними процесами;
- ситуаційне управління;
- технічна та медична діагностика;
- прогнозування часових рядів:
- розпізнавання образів;
- багатофакторний аналіз.

На відміну від моделей типу «чорний ящик», нечіткі бази знань прозорі, їх структура змістовно інтерпретується в термінах, зрозумілих не тільки розробникам з високою математичною кваліфікацією, а і замовникам — лікарям, економістам, менеджерам.

Інтерпретабельність нечітких моделей — одна з головних переваг, завдяки якій нечіткі технології успішно конкурують з іншими методами, особливо для тих прикладних завдань, де змістовна інтерпретація важливіша за точність моделювання.

Об’єктом дослідження є структурна ідентифікація багатофакторних залежностей.

Предметом дослідження є методи відбору правил до нечіткої бази знань для ідентифікація багатофакторних залежностей.

Метою дослідження є підвищення точності та компактності нечітких баз знань за рахунок використання суперечливих правил.

Для досягнення мети поставлено та вирішено такі завдання дослідження:

1) Досліджено предметну галузь для поглиблення теоретичних знань.

2) Розглянуто існуючі моделі та алгоритми вирішення задачі відбору правил, з виділенням їх переваг та недоліків, для створення власних.

3) Створено модель та на її основі розроблені власні алгоритми відбору правил нечітких баз знань, які використовують суперечливі правила.

4) Проведені обчислювальні експерименти головною метою яких було порівняння кривих навчання нечітких баз з та без суперечливих правил, як результат, доведення застосування суперечливих правил в нечітких базах знань.

Наукова новизна дослідження

Запропоновано метод відбору правил нечіткої бази знань Мамдані, який на відмінну від інших, дозволяє використовувати суперечливі правилами з сусідніми лінгвістичними значеннями консеквентів.

Практичне значення отриманих результатів полягає в тому, що рахунок використання суперечливих правил вдається отримати нечіткі бази знань Мамдані з кращими показниками точності та компактності.

Апробація результатів дослідження здійснена на конференціях III Всеукраїнській науково-практичні конференції для студентів, аспірантів та молодих вчених «Прикладні інформаційні технології»(ПІТ-2022) та III Всеукраїнській науково-практичні конференції «Комп’ютерні технології обробки даних»(КТОД 2022)

Публікації: за матеріалами роботи опубліковано 2 тези доповіді на конференціях [1, 2] та підготовлена стаття для журналу «Вісник Донецького національного університету імені Василя Стуса: Серія «Комп’ютерні науки та кібер-фізичні системи»[3].

Структура роботи

Робота складається зі вступу, чотирьох розділів та висновків до них, списку використаних джерел та додатків.

У першому розділі роботи аналізується предметна область, зокрема нечітка логіка та нечіткі бази знань, здійснюється огляд методів селекції правил баз знань Мамдані для ідентифікації багатофакторних залежностей, визначаються їх переваги та недоліки. Формулюється ідея дослідження та деталізуються подальші завдання роботи.

У другому розділі наводяться власні ідеї, методи та підходи модифікації, покращення розглянутих в попередньому розділі алгоритмів, відбору правил нечітких баз знань. Виділені їхні переваги та недоліки.

У третьому розділі демонструється як здійснюється нечітка ідентифікація в системі Matlab. Наведено архітектуру системи, які модулі наявні в системі та які функції вони виконують. Коротко описано процес розробки.

У четвертому розділі аналізуються обчислювальні експерименти із синтезу нечітких баз знань Мамдані з суперечливими правилами, головною метою яких є порівняння кривих навчання нечітких баз з та без суперечливих правил, як результат, доведення чи спростування доречності застосування суперечливих правил в нечітких базах знань.

Кваліфікаційна робота включає в себе 71 сторіноку, 43 рисунки і список літератури з 50 джерел.

РОЗДІЛ 1
ОГЛЯД СТАНУ ПИТАННЯ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

У розділі аналізується предметна область, зокрема нечітка логіка та нечіткі бази знань, здійснюється огляд методів селекції правил баз знань Мамдані для ідентифікації багатофакторних залежностей, визначаються їх переваги та недоліки. Формулюється ідея дослідження та деталізуються подальші завдання роботи.

1.1 Основні положення теорії нечітких множин

Доволі часто виникають труднощі в ідентифікації залежностей практичних задач, як правило, вони обумовлені відсутністю достатнього обсягу точних експериментальних даних або наявністю результатів спостережень в формі нечітких знань та лінгвістичних експертних оцінок [4, 5]. Один із шляхів подолання цих труднощів полягає у застосуванні нечіткої ідентифікації, тобто методів побудови моделей на основі теорії нечітких множин та нечіткої логіки. Найчастіше нечітка ідентифікація проводиться коли модель залежності являє собою нечітку базу знань.

Основою теорії нечітких множин є ідея про те, що елементи з певної множини, які володіють деякою спільною властивістю, можуть володіти нею з різним ступенем. За такого підходу висловлювання про те, що елемент належить множині втрачає сенс, тому що необхідно вказати наскільки сильно або з яким ступенем елемент задовольняє властивостям даної множини.

Базуючись на [6, 7, 8, 9, 10] наведемо основні положення теорії нечітких множин, які необхідні для подальшого викладення матеріалу.

Нечіткою множиною $\widetilde{A}$ на універсальній множині U називається сукупність пар $(\mu_A(u), u)$, де $\mu_A(u)$ – ступінь належності елемента $u \in U$ до нечіткої множини $\widetilde{A}$. Ступінь належності – це число з діапазону [0, 1]. Чим більший ступінь належності, тим сильніше елемент універсальної множини відповідає властивостям нечіткої множини.

Функцією належності називається така функція, яка дозволяє обчислити ступінь належності довільного елемента універсальної множини до нечіткої множини.

Якщо універсальна множина складається з кінцевого числа елементів $U = \{u_1, u_2, ..., u_k\}$, тоді нечітка множина $\widetilde{A}$ записується так: $\widetilde{A} = \left(\frac{\mu_A(u_1)}{u_1}, \frac{\mu_A(u_2)}{u_2}, ..., \frac{\mu_A(u_k)}{u_k} \right)$. У випадку неперервної універсальної множини U використовують запис $\widetilde{A} = \int_u \mu_A(u)/u$, де знак $\int$ означає сукупність пар $\mu_A(u)$ та u.

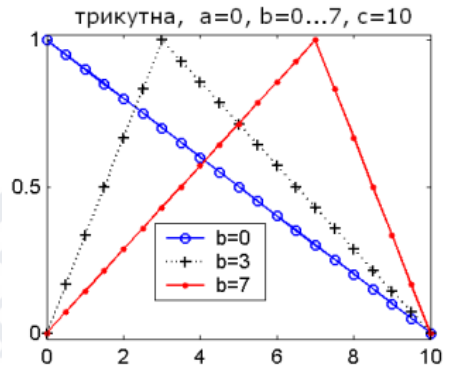
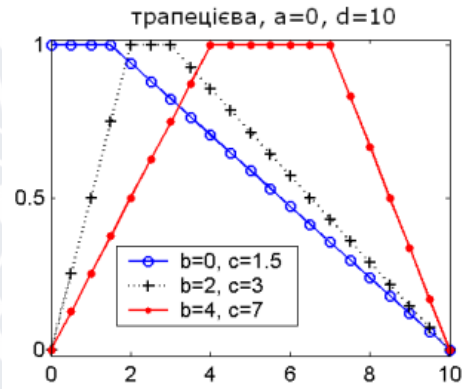
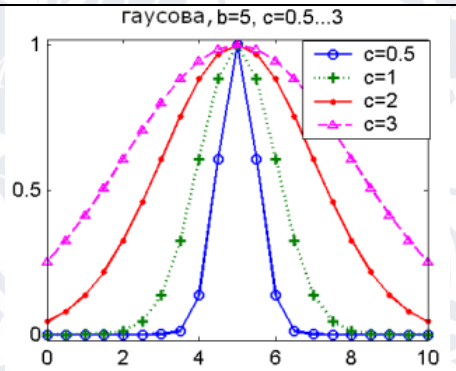
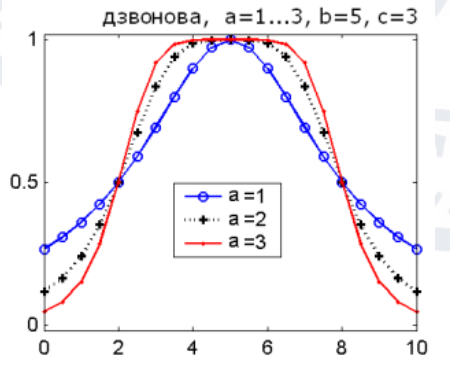
Носієм нечіткої множини називається множина усіх елементів універсальної множини, які мають ненульовий ступінь належності. Носій нечіткої множини $\widetilde{A}$ позначається $\text{supp}(\widetilde{A})$.

Ядром нечіткої множини називається множина усіх елементів універсальної множини, ступінь належності яких дорівнює 1. Ядро нечіткої множини $\widetilde{A}$ позначається $\text{core}(\widetilde{A})$.

Лінгвістичною змінною називається змінна, значеннями якої є слова або словосполучення природної мови. Множина усіх можливих значень лінгвістичної змінної називається терм-множиною. Елемент терм-множини називається термом. В теорії нечітких множин терм задається функцією належності.

Параметричні функції належності [12] зазвичай мають 2, 3 або 4 параметри. Існує багато типів параметричних функцій належностей, найбільш розповсюдженими серед яких є трикутна, трапецієва, гаусова та дзвонова. Аналітичні вирази цих функцій належностей та їх графічне зображення зведені в табл. 1.1.

Таблиця 1.1 – Популярні параметричні функцій належності [11]

Функції належності	Аналітичний вираз
	$\mu(x)=\begin{cases}0, & \text{якщо } y<a\\ \dfrac{x-a}{b-a}, & \text{якщо } a\le x<b\\ \dfrac{c-x}{c-b}, & \text{якщо } b\le x\le c\\ 0, & \text{якщо } x>c\end{cases},$ де b – координата максимуму; (a, c) – носій нечіткої множини $\tilde{x}$.
	$\mu(x)=\begin{cases}0, & \text{якщо } x<a\\ \dfrac{x-a}{b-a}, & \text{якщо } a\le x<b\\ 1, & \text{якщо } b\le x<c\\ \dfrac{d-x}{d-c}, & \text{якщо } c\le x\le d\\ 0, & \text{якщо } x>d\end{cases},$ де $[b, c]$ – ядро нечіткої множини $\tilde{x}$; (a, d) – носій нечіткої множини $\tilde{x}$.
	$\mu(x)=\exp\left(-\dfrac{(x-b)^2}{2c^2}\right),$ де b – координата максимуму; c – коефіцієнт концентрації.
	$\mu(x)=\dfrac{1}{1+\left\lvert\dfrac{x-b}{c}\right\rvert^{2a}},$ де a – коефіцієнт крутизни; b – координата максимуму; c – коефіцієнт концентрації.

Позначимо нечіткі логічні змінні через $\widetilde{A}$ і $\widetilde{B}$, а функції належності, що задають істинності значення цих змінних через $\mu_{\widetilde{A}}(u)$ і $\mu_{\widetilde{B}}(u)$, $u \in [0,1]$.

Нечітка логіка – це різновид багатозначної логіки, в якій значення істинності задаються лінгвістичними змінними або такими термами лінгвістичної змінної "істинність" як: "дуже істинно", "майже істинно", "трохи хибно" тощо. Ці лінгвістичні значення істинності описуються нечіткими множинами. Правила виконання нечітких логічних операцій отримують з булевих логічних операцій за допомогою принципу нечіткого узагальнення.

Нечіткі логічні операції ТА ($\wedge$) (1.1), АБО ($\vee$) (1.2), НІ ($\neg$) (1.3) та імплікація ($\Rightarrow$) (1.4) виконуються за такими правилами:

$$\mu_{A \wedge B}(u) = \min\left(\mu_A(u), \mu_B(u)\right), \tag{1.1}$$

$$\mu_{A \vee B}(u) = \max\left(\mu_A(u), \mu_B(u)\right), \tag{1.2}$$

$$\mu_{\overline{A}}(u) = 1 - \mu_A(u), \tag{1.3}$$

$$\mu_{A \Rightarrow B}(u) = \max\left(1 - \mu_A(u), \mu_B(u)\right). \tag{1.4}$$

1.2 Нечітка база знань. База знань Мамдані

Нечіткою базою знань називається сукупність нечітких правил <Якщо – тоді>, які описують певну предметну область. «Якщо» – частина правила називається антецедентом або посилкою, а «тоді» – частина правила – консеквентом або висновком [13, 14]. В даній роботі розглядається база знань Мамдані, в якій антецеденти і консеквенти задано нечіткими множинами. Цю базу знань можна трактувати як розбиття факторного простору на зони з нечіткими межами, в кожній з яких функція відклику приймає нечітке значення. Кількість нечітких зон дорівнює числу правил. З [15, 16, 17]Нечітку базу знань Мамдані, що описує залежність $y = f(x_1, x_2, \ldots, x_n)$, представимо таким чином:

$$\begin{aligned}&\text{Якщо}\left(x_1=\tilde{a}_{1j}\ \text{ та }\ x_2=\tilde{a}_{2j}\ \text{ та ... та }\ x_n=\tilde{a}_{nj}\ \text{ з вагою } w_j\right),\\&\text{тоді}\quad y=\tilde{d}_j,\quad j=\overline{1,N},\end{aligned}\tag{1.5}$$

де $\tilde{d}_j$ – нечітке значення, яке обирається з терм-множини $\{\tilde{l}_1, \tilde{l}_2, ..., \tilde{l}_m\}$, кожен елемент якої представлено нечіткою множиною $\tilde{l}_s = \int\limits_{y\in[\underline{y},\,\overline{y}]} \mu_{l_s}(y)/y$, $s=\overline{1,m}$;

$w_j \in [0,1]$ - ваговий коефіцієнт, який віддзеркалює впевненість експерта в адекватності j-го правила;

N - кількість правил в базі знань.

Нечітку базу знань зручно подавати у формі табл. 1.2.

Таблиця 1.2 - Нечітка база знань Мамдані в табличній формі [6]

Якщо				Тоді
x_1	x_2	…	x_n	y
$\tilde{a}_{11}$	$\tilde{a}_{21}$	…	$\tilde{a}_{n1}$	d_1
$\tilde{a}_{12}$	$\tilde{a}_{22}$	…	$\tilde{a}_{n2}$	d_2
...				
$\tilde{a}_{1N}$	$\tilde{a}_{2N}$	…	$\tilde{a}_{nN}$	d_N

Логічне виведення за нечіткою базою знань (1.5) здійснюють за схемою з рис. 1.1. Спочатку для поточного вхідного вектора $X^*=\left(x_1^*, x_2^*, \ldots, x_n^*\right)$ розраховують ступінь виконання антецедента *j*-го правила:

$$\mu_j(X^*)=\mu_j(x_1^*)\ \wedge\ \mu_j(x_2^*)\ \wedge ... \wedge\ \mu_j(x_n^*)\ ,\ j=\overline{1,N}\,,\tag{1.6}$$

де $\wedge$ – t-норма, яку в алгоритмі Мамдані зазвичай реалізують операцією мінімуму.

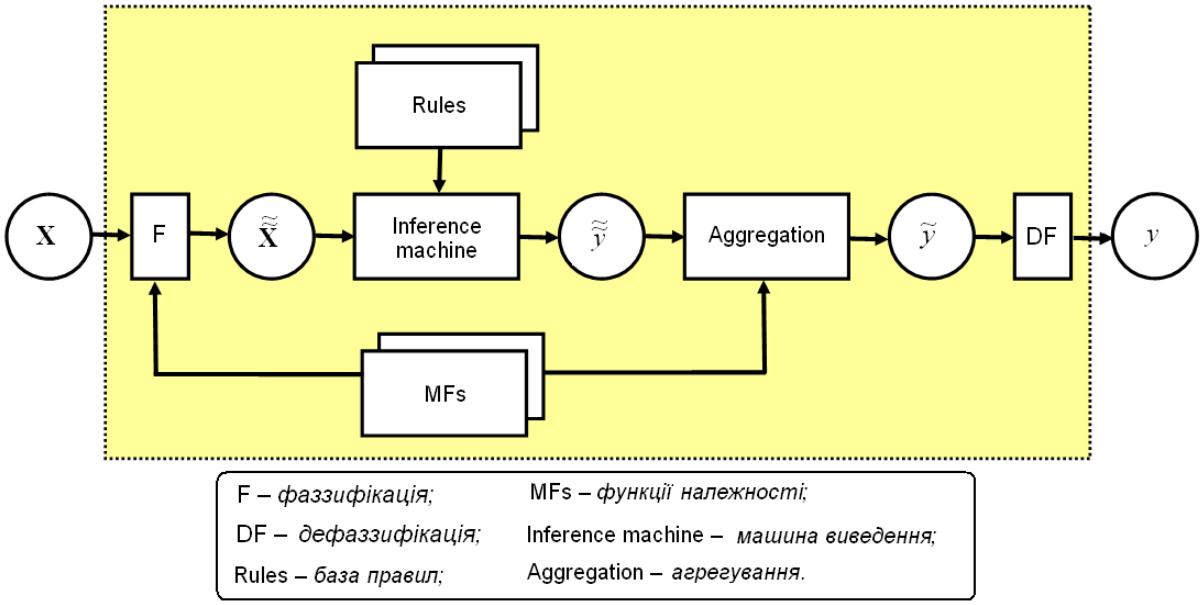


Рисунок 1.1 – Логічне виведення за нечіткою базою знань Мамдані [6]

Результат логічного виведення можна записати у формі такої бінечіткої множини:

$$\tilde{\tilde{y}}^* = \left(\frac{\mu_{d_1}(X^*)}{\tilde{d}_1}, \frac{\mu_{d_2}(X^*)}{\tilde{d}_2}, ..., \frac{\mu_{d_N}(X^*)}{\tilde{d}_N} \right),$$

особливістю якої є те, що елементами її носія є нечіткі множини $\tilde{d}_1, \tilde{d}_2, ..., \tilde{d}_N$. Для перетворення $\tilde{\tilde{y}}^*$ в звичайну нечітку множину виконаємо такі дії. Спочатку представимо результат виведення за *j*-им правилом бази знань у формі такої нечіткої множина:

$$\tilde{d}_j^* = \text{imp}\left(\tilde{d}_j,\ \mu_j(X^*)\right),\ j = \overline{1, N}, \qquad (1.7),$$

де imp – позначає імплікацію, яку реалізують операцією мінімуму.

Геометричною інтерпретацією імплікації є зрізання графіка функції належності $\mu_{d_j}(y)$ по рівню $\mu_j(X^*)$, що математично запишемо так:

$$\tilde{d}_j^* = \int_{y \in [\underline{y}, \bar{y}]} \min\left(\mu_j(X^*),\ \mu_{d_j}(y)\right) / y\,.$$

Результат виведення за усіма правилами знаходять агрегуванням нечітких множин (1.7):

$$\tilde{y}^* = \text{agg}\left(\tilde{d}_1^*, \tilde{d}_2^*, ..., \tilde{d}_N^*\right), \tag{1.8}$$

де agg – агрегування нечітких множин, яке реалізують операцією максимуму.

Ілюстрацією цієї формули є рис. 1.2, де здійснюється агрегування трьох нечітких множин.

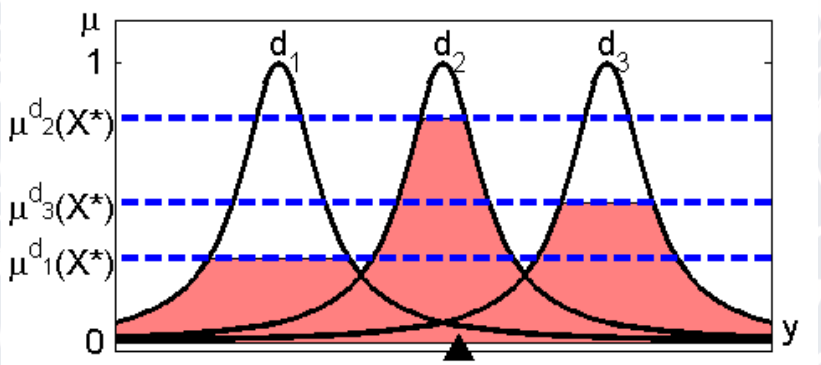


Рис. 1.2 - Імплікація, агрегування та дефазифікація в алгоритмі Мамдані [6]

Чітке значення виходу y^*, яке відповідає вхідному вектору X^*, визначається через дефазифікацію нечіткої множини $\tilde{y}^*$ за центром тяжіння.

1.3 Огляд методів відбору правил нечіткої бази знань

Ідентифікація багатофакторних залежностей за допомогою нечіткої бази знань Мамдані здійснюється у 2 етапи. На першому етапі – на етапі структурної ідентифікації формується у деякий спосіб нечітка база знань – обираються терм-множини, задаються їх функції належності та синтезуються правила бази знань. На другому етапі – на етапі параметричної ідентифікації налаштовуються параметри функцій належності [18]. Наше дослідження стосується першого етапу – етапу структурної ідентифікації, а саме відбору правил до нечіткої бази знань.

Відбір правил нечіткої бази знань можна звести до оптимізаційної задачі про рюкзак. Правилу бази знань відповідає предмет, який може потрапити до рюкзака, точності бази знань – корисність рюкзака, а кількості правил – сумарна об’єм обраних предметів [19, 20, 21]. Відмінність між задачами полягає в різних типах функції корисності, яка є лінійною в задачі про рюкзак та нелінійною в задачі відбору правил бази знань. Аналогічно до класичних постановок задачі про рюкзак сформовано й задачі відбору правил. Задача відбору правил, як і задача про рюкзак, є NP-повною. Відповідно алгоритм точного розв’язання цієї задачі матиме експоненціальну обчислювальну складність, і тому буде прийнятним лише за невеликої кількості правил-кандидатів.

Під час розробки нечітких баз для вирішення практичних задач намагаються забезпечити коректний баланс між компактністю та точністю. Необхідною умовою такого балансу є потрапляння бази знань на парето-фронт у координатах “складність моделі – точність моделі”. Ключовими роботами за цією тематикою є статті [22, 23] з формування множини баз знань нечіткого класифікатора, які належать парето-фронту недомінантних альтернатив у координатах “кількість правил – безпомилковість”. Для цього застосовують задачі оптимізації з метою: 1) максимізації безпомилковості за обмеженої кількості правил; 2) мінімізації кількості правил за деякого рівня безпомилковості; 3) мінімізації інтегрального критерію якості бази знань у формі лінійної згортки безпомилковості та кількості правил [23] або безпомилковості, кількості правил та сумарної довжини антецедентів правил [24]. Оцінка якості баз знань [25]. Для отримання парето-фронту оптимізацію проводять багаторазово за різних граничних значень в обмеженнях задач 1 і 2 та вагових коефіцієнтів цільової функції в задачі 3. Аналогічні підходи застосовують, обираючи правила нечітких баз знань для об’єктів з неперервним виходом [26].

Оскільки алгоритми точного розв’язання вказаних задач мають експоненціальну складність, вони мають прийнятне рішення лише за

невеликої кількості правил-кандидатів. На практиці для розв'язання цієї задачі зазвичай застосовують генетичні алгоритми [26, 27, 28, 29]. Для задач малої розмірності можна використовувати повний перебір, як це показано в роботі [30]. Для задач дуже великої розмірності вибір правил можна здійснити за жадібним алгоритмом, який дозволяє дуже швидко знайти раціональні розв'язки [6, 28], тобто сформувати більш-менш прийнятний набір правил. Для покращення розв'язків застосовують пряму та зворотні схеми жадібного алгоритму – за прямою схемою починають з порожньої бази знань і ітераційно додають по одному найкращому правилу. За зворотної схеми починають з повною бази знань, з якої на кожному кроці вилучають найгірше правило.

1.4 Синтез правил нечітких баз знань, алгоритм Ванга-Менделя

Один з найбільш відомих алгоритмів синтезу правил нечітких баз знань – алгоритм Ванга-Менделя. Згідно [31] метод Ванга-Менделя полягає в генерації нечітких правил із заданих пар даних.

Припускається, що наявний набір даних вхід-вихід виду:

$$(x_1^{(1)}, x_2^{(1)}; y^{(1)}), (x_1^{(2)}, x_2^{(2)}; y^{(2)}), \ldots, (x_1^{(n)}, x_2^{(n)}; y^{(n)}), \quad (1.9)$$

де x_1, x_2 - вхідні значення;
y - вихідні значення;
n - кількість пар вхідних даних.

В даному випадку наводиться система з двома входами та одним виходом, для пояснення основних ідей методу; розширення до загальних випадків із кількома входами та кількома виходами є доволі простою.

В загальному алгоритм синтезу правил нечіткої бази знань складається з таких етапів:

1. Розбиття факторного простору на зони з нечіткими межами, в кожній з яких функція відклику приймає нечітке значення (розбиття вхідних та вихідних значень на нечіткі зони).

2. Генерація нечітких правил з експериментальних даних (пари вхід-вихід (1.9))
3. Призначення ступеня кожному з правил отриманим з набору даних, та усунення суперечливих правил.

Спершу потрібно розділити простір введення та виведення на нечіткі області. Припустимо, що інтервали належності x_1, x_2, та у лежать в $[x_1^-, x_1^+]$, $[x_2^-, x_2^+]$ і $[y^-, y^+]$, відповідно, де «інтервали належності» змінної означає, що найбільш ймовірно, ця змінна буде лежати в цьому інтервалі. З рис. 1.3 видно, що для змінної x_1, розбиття здійснено на 5 областей, для змінної x_2 розбиття здійснено на 7 областей та для *y* на 5 областей. Також допускаються і інші варіанти розбиття.

Далі потрібно згенерувати нечіткі правила із заданих пар даних. Для цього слід визначити ступені належності всіх змінних $x_1^{(i)}$, $x_2^{(i)}$ та $y^{(i)}$ до нечітких множин.

Для прикладу на рис. 1.3 $x_1^{(1)}$ належить до терму B1 зі ступенем належності 0,8, 0,2 до B2 і нульову ступінь для інших множин. Так само $x_2^{(2)}$ має одиничний ступінь належності до множини CE, а до всіх інших нульову.

Після цього, для кожної зміни потрібно відібрати множини з максимальним ступенем належності. Для прикладу див. рис. 1.3, $x_1^{(1)}$ має максимальну ступінь належності до множини B1.

На кінець потрібно сформувати правило з функцій належності з максимальним ступенем, наведені кроки потрібно повторити для кожної пари вхід-вихід.

($x_1^{(1)}$, $x_2^{(1)}$, $y^{(1)}$) → [$x_1^{(1)}$(0,8 B1, max), $x_2^{(1)}$(0,7 S1, max), $y^{(1)}$ (0,9 CE, max)] → Правило 1 : Якщо x_1 це B1 та x_2 це S1, то y є CE.

$(x_1^{(2)}, x_2^{(2)}, y^{(2)}) \rightarrow [x_1^{(2)}$(0,6 B1, max), $x_2^{(2)}$(1 SE, max), $y^{(2)}$ (0,7 B1, max)] → Правило 2 : Якщо x_1 це B1 та x_2 це SE, то у є B1.

На кінець потрібно присвоїти ступінь кожному правилу. Оскільки зазвичай існує багато пар даних, і кожна пара даних генерує одне правило, дуже ймовірно, що будуть згенеровані конфліктні правила, тобто правила, які мають однакову частину «якщо», але різну частину «то». Один зі способів вирішення цього конфлікту – це призначення ступеня кожному правилу, створеному з пар даних, і приймати лише те правило з конфліктної групи, яке має максимальну ступінь. Ступінь правила визначається як добуток ступенів його складових, що власне створює це правило. Таким чином не тільки вирішується конфліктна проблема, але й кількість правил значно зменшується.

Для прикладу маємо правила з однаковою умовою «якщо» та різною умовою «то»:

$(x_1^{(a)}, x_2^{(a)}, y^{(a)}) \rightarrow [x_1^{(a)}$(0,5 S1, max), $x_2^{(a)}$(0,6 B3, max), $y^{(a)}$ (0,7 CE, max)] → Правило a : Якщо x_1 це B1 та x_2 це S1, то у є CE.

$(x_1^{(b)}, x_2^{(b)}, y^{(b)}) \rightarrow [x_1^{(b)}$(0,9 S1, max), $x_2^{(b)}$(1 B3, max), $y^{(b)}$ (0,8 B1, max)] → Правило b : Якщо x_1 це B1 та x_2 це SE, то у є B1.

D (Правило a) = $m_{S1}(x_1)\, m_{B3}(x_2)\, m_{CE}$(y) = 0,5 × 0,6 × 0,7 = 0,21.

D (Правило b) = $m_{S1}(x_1)\, m_{B3}(x_2)\, m_{B1}$(y) = 0,9 × 1 × 0,8 = 0,72.

Отже, в базу знань потрапить правило b, оскільки воно має більший ступінь належності, а це означає, що йому можна більше довіряти

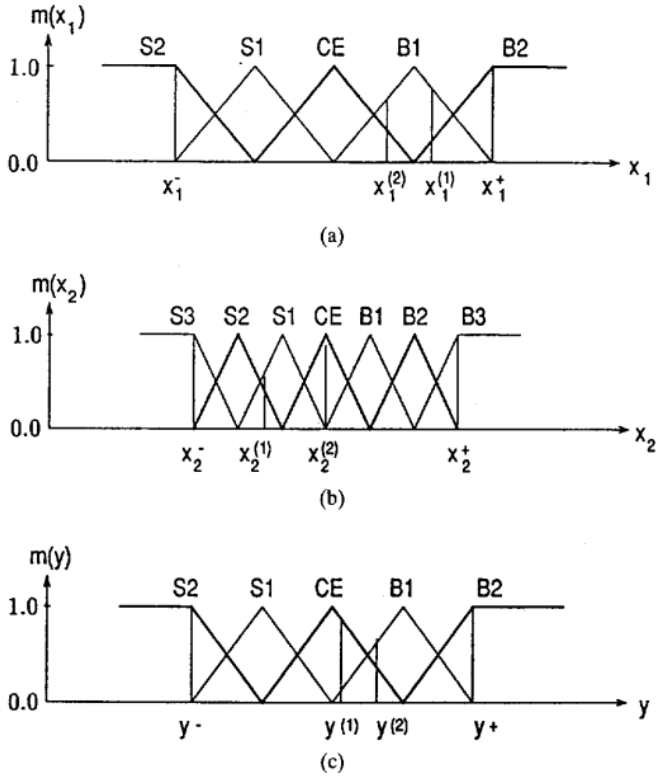


Рисунок 1.3 - Поділ вхідного та вихідного просторів на нечіткі області та відповідні функції належності [31]

Даний алгоритм є доволі простий та наочний, він дозволяє швидко та з прийнятною точністю синтезувати правила нечіткої бази знань. Головним недоліком цього методу є те, що синтез правил відбувається за рахунок однієї найсильнішої пари вхід-вихід, тобто, багато даних ніяк не враховуються при синтезі правил нечітких баз знань.

1.5 Ідея дослідження та деталізація завдань

Для усіх проаналізованих до цього методів передбачається, що існує список правил-кандидатів, з якого і потрібно обрати деяку множину правил. При цьому вважається, що правила в сформованій базі знань не будуть суперечливими. З [1, 3] суперечливими розуміються правила з однаковими антецедентами та різними консеквентами, наприклад:

Якщо $(x_1 = низький$ та $x_2 = середній)$, тоді $y = середній$;

Якщо $(x_1 = низький$ та $x_2 = середній)$, тоді $y = високий$.

Разом з тим, якщо в базі знань буде така пара суперечливих правил, тоді результат логічного виведення за ними буде між *середній* та *високим*, тобто відповідатиме терму *вище середнього*, якого немає в терм-множині вихідної змінної *y*. Таким чином, застосування таких суперечливих правил підвищує «роздільну здатність» опису залежності, яку моделює база знань. Відповідно, з'являються шанси підвищити точність нечіткої бази знань за рахунок використання суперечливих правил.

Для виконання ідеї дослідження потрібно вирішити ряд наступних завдань:

1) З розглянутих методів і алгоритмів, синтезу та відбору правил нечітких баз знань, виділити їхні переваги та недоліки, для подальшої розробки.
2) Створити модель та базуючись на ній розробити власні алгоритми відбору правил нечітких баз знань, які використовують суперечливі правила.
3) Провести обчислювальні експерименти головною метою яких є проведення обчислювальних експериментів із синтезу нечітких баз знань Мамдані з суперечливими правилами, та порівняння кривих навчання нечітких баз з та без суперечливих правил.
4) Підбити підсумки та встановити доцільність застосування суперечливих правил в нечітких базах знань.

Висновки до розділу 1

Виявлено, що існуючі методи та алгоритми синтезу і відбору правил нечітких баз знань мають ряд недоліків.

Встановлено, що деякі з алгоритмів мають експоненційну складність, або близьку до неї, тобто працюють дуже довго, в інших точність не близька до прийнятної, також розглянуті алгоритми синтезують нечіткі бази знань великої розмірності, що також не дуже добре, оскільки їх використання та інтерпретація ускладняється.

Запропоновано власний підхід з застосуванням суперечливих правил в відборі правил нечітких баз знань. Адже це підвищить «роздільну здатність» опису залежності, яку моделює база знань. Відповідно, з’являються шанси підвищити точність нечіткої бази знань за рахунок використання суперечливих правил, та скоротити кількість правил, як наслідок зробити базу знань компактнішою.

РОЗДІЛ 2
МОДЕЛІ ТА АЛГОРИТМИ ВИРІШЕННЯ ЗАДАЧІ ВІДБОРУ ПРАВИЛ НЕЧІТКИХ БАЗ ЗНАНЬ

В даному розділі наводяться власні ідеї, методи та підходи модифікації, покращення розглянутих в попередньому розділі алгоритмів, відбору правил нечітких баз знань. Виділені їхні переваги та недоліки.

2.1 Модифікація алгоритму Ванга-Менделя – «сума голосів»

В класичному алгоритмі Ванга-Менделя генерація правил відбувається шляхом відбору правил з пар вхід-вихід – які активують терми з максимальними ступнями належності. Тобто, перемагає те правило яке має максимальну ступінь належності. Такий підхід ще називають: «переможець забирає все», і він не завжди є доречним. Адже багато даних ніяк не враховуються при синтезі правил нечітких баз знань. Згідно [32, 33, 34, 35, 36] існує багато варіантів модифікації цього алгоритму. Один з варіантів модифікації – сумувати ступені належності всіх пар вхідних даних які активують однакові правила. І вже далі з сум вибирати правила з максимальними значеннями.

Тут також, дуже ймовірно, що будуть згенеровані конфліктні правила. Для вирішення цієї проблеми потрібно приймати лише те правило з конфліктної групи, яке має максимальне значення суми, що власне створює це правило. Таким чином не тільки вирішується проблема конфліктних правил, а їх кількість значно зменшується, коротко етапи подано на рис. 2.1.

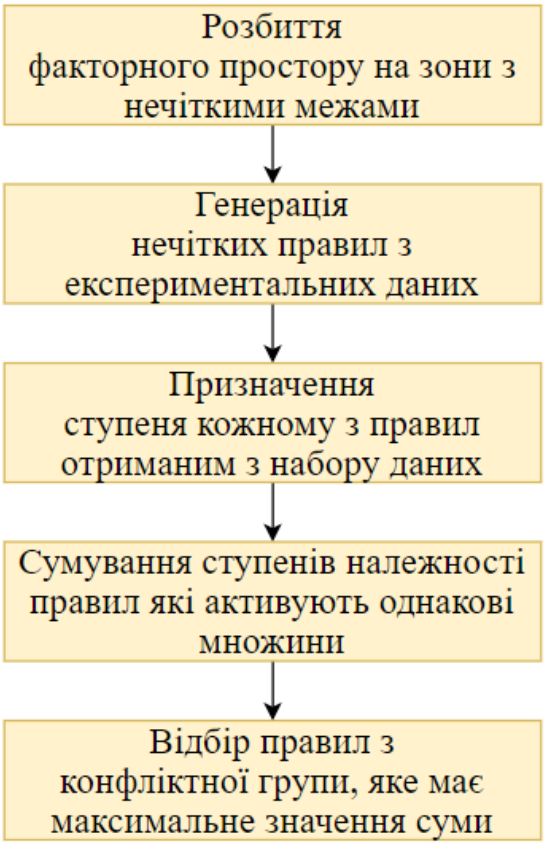


Рисунок 2.1 – Порядок дій модифікації алгоритму Ванга-Менделя «сума голосів»

Наприклад: маємо 5 пар вхід-вихід які активують відповідні множини з певними ступенями належності, як показано в табл. 2.1. З неї видно, що всі пари однакові в частині висновку – частині у. В класичному алгоритмі Ванга-Менделя до бази знань потрапило б правило: «Якщо x_1 Високий та x_2 це Високий, то у Низький», оскільки саме це правило має найбільший ступінь належності 0,48, хоча воно активується лише 1 раз.

Тут ми не врахували що правило: «Якщо x_1 Низький та x_2 Низький, то у Низький», має чималу кількість активацій з достатнім ступенем належності, і можливо краще було б згенерувати саме його.

Таблиця 2.1 – Належність та ступінь до множин пар вхід-вихід

	x1		x2		y		
Вхід-вихід	Терм-множина	Ступінь належності	Терм-множина	Ступінь належності	Терм-множина	Ступінь належності	Ступінь правила
1	Низький	0,7	Низький	0,6	Низький	0,8	0,336
2	Низький	0,6	Низький	0,5	Низький	1	0,3
3	Низький	0,65	Низький	0,6	Низький	0,7	0,273
4	Середній	0,6	Середній	0,7	Низький	0,4	0,168
5	Високий	0,8	Високий	0,75	Низький	0,8	0,48

Запропонований метод «сум голосів» враховує всі виклики. Тут ми обираємо з сум всіх ступенів, найбільше.

В даному випадку буде обрано правило: «Якщо x_1 Низький та x_2 Низький, то у Низький» оскільки сума степенів активації найбільша і складає 0,909.

Даний алгоритм, володіє ключовою перевагою над класичним алгоритмом Ванга-Менделя, оскільки враховуються всі активації вхідних пар даних, як наслідок прогнозована точність баз знань отриманих алгоритмом «сум голосів», краща за бази знань отримані класичним алгоритмом Ванга-Менделя.

2.2 Модифікація алгоритму Ванга-Менделя – «головний конкурент»

Ще один з варіантів модифікації алгоритму Ванга-Менделя – це до бази знань вносити не лише правило переможець за максимальними ступнями належності, а й вносити правило яке найближче конкурує з ним в частині консеквента – висновка, тобто, «головний конкурент», згідно роботи [2]. Подібні модифікації з комбінацією правил та змінних також розглядались у роботах [37, 38, 39].

Як і в класичному алгоритмі Ванга-Менделя спершу потрібно розділити простір введення та виведення на нечіткі області.

Далі потрібно визначити ступені належності всіх змінних $x_1^{(i)}$, $x_2^{(i)}$ та $y^{(i)}$ до нечітких множин.

Для прикладу на рис. 1.3 $x_1^{(1)}$ належить до терму B1 зі ступенем належності 0,8, 0,2 до B2 і нульову ступінь для інших термів. Так само $x_2^{(2)}$ має одиничний ступінь належності до множини CE, а до всіх інших нульову.

Далі, для кожної змінної потрібно відібрати множини з максимальним ступенем належності. Для прикладу див. рис. 1.3, $x_1^{(1)}$ має максимальну ступінь належності до множини B1, $x_2^{(1)}$ має максимальну ступінь належності до множини S1, та $y^{(1)}$ має максимальну ступінь належності до множини SE. Власне з цих максимальних ступенів належності до множин і формуються відповідні правила, наведені кроки потрібно повторити для кожної пари вхід-вихід.

($x_1^{(1)}$, $x_2^{(1)}$, $y^{(1)}$) → [$x_1^{(1)}$(0,8 B1, max), $x_2^{(1)}$(0,7 S1, max), $y^{(1)}$ (0,9 CE, max)]→

Правило 1 : Якщо x_1 це B1 та x_2 це S1, то y є CE.

($x_1^{(2)}$, $x_2^{(2)}$, $y^{(2)}$) → [$x_1^{(2)}$(0,6 B1, max), $x_2^{(2)}$(1 SE, max), $y^{(2)}$ (0,7 B1, max)]→

Правило 2 : Якщо x_1 це B1 та x_2 це SE, то y є B1.

На кінець навмисно накладають обмеження, щоб в базу знань не потрапило конфліктного правила. Шляхом призначення ступеня кожному правилу, створеному з пар даних, і відбором того правила з конфліктної групи, яке має максимальну ступінь. Ступінь правила визначається як добуток ступенів його складових, що власне створює це правило. Таким чином отримана база знань виходить без конфліктних правил.

Ми ж допускаємо відбір суперечливих правил, і для кожного відібраного правила в попередньому кроці, ми формулюємо правила такі ж в частині

антецедента та множиною яка має наступний за величиною ступінь активації, в частині консеквента. Всі наведені вище кроки коротко подано на рис. 2.2.

Для прикладу див. рис. 1.3, y^1 має максимальну ступінь належності до множини CE 0,9 та найближчу конкуруючу множину B1 з ступенем 0,1, а y^2 має максимальну ступінь належності до множини B1 0,7 та найближчу конкуруючу множину CE з ступенем 0,45.

На кінець потрібно сформувати правила з функцій належності з максимальним ступенем та конкурентом в частині висновка, наведені кроки потрібно повторити для кожної пари вхід-вихід:

($x_1^{(1)}$, $x_2^{(1)}$, $y^{(1)}$) → [$x_1^{(1)}$(0,8 B1, max), $x_2^{(1)}$(0,7 S1, max), $y^{(1)}$ (0,9 CE, max)] → Правило 1 : Якщо x_1 це B1 та x_2 це S1, то у є CE.

($x_1^{(1)}$, $x_2^{(1)}$, $y^{(1)}$) → [$x_1^{(1)}$(0,8 B1, max), $x_2^{(1)}$(0,7 S1, max), $y^{(1)}$ (0,1 B1, max)] → Правило 2 (Правило конкурент) : Якщо x_1 це B1 та x_2 це S1, то у є B1.

($x_1^{(2)}$, $x_2^{(2)}$, $y^{(2)}$) → [$x_1^{(2)}$(0,6 B1, max), $x_2^{(2)}$(1 SE, max), $y^{(2)}$ (0,7 B1, max)] → Правило 3 : Якщо x_1 це B1 та x_2 це SE, то у є B1.

($x_1^{(2)}$, $x_2^{(2)}$, $y^{(2)}$) → [$x_1^{(2)}$(0,6 B1, max), $x_2^{(2)}$(1 SE, max), $y^{(2)}$ (0,45 CE, max)] → Правило 4 (Правило конкурент) : Якщо x_1 це B1 та x_2 це SE, то у є CE.

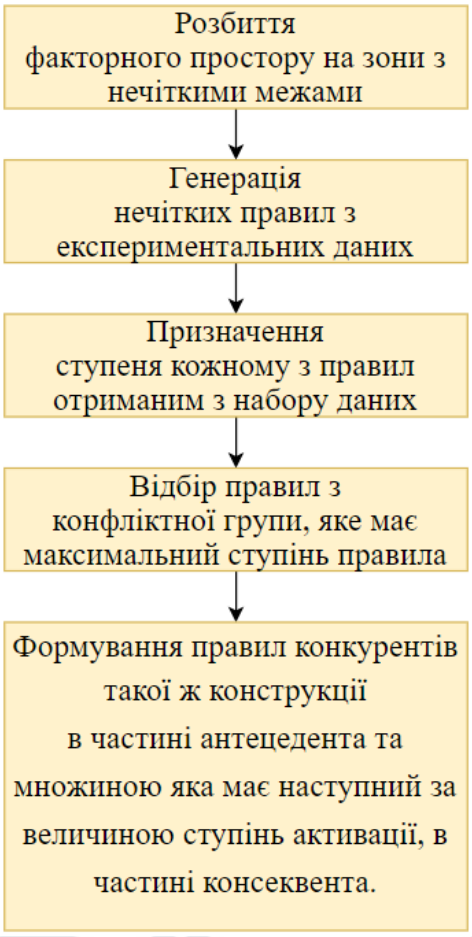


Рисунок 2.2 - Порядок дій модифікації алгоритму Ванга-Менделя «сума голосів»

Даний алгоритм «головного конкурента» має єдину відмінність від класичного алгоритму Ванга-Менделя, в ньому дозволяється, і навіть задовольняється, використання суперечливих правил за принципом «головного конкурента». В якому до бази знань вноситься не лише правило переможець з максимальним ступенем активації, але і друге за рангом правило з тим самим антецедентом. Таким чином, формується база знань із суперечливими правилами, застосування яких підвищує «роздільну здатність» опису залежності, яку моделює база знань. Відповідно, з'являються шанси підвищити точність нечіткої бази знань за рахунок використання таких правил.

2.3 Синтез суперечливих нечітких баз знань з підмножини суперечливих, метод найближчого сусіда

Метод полягає в формуванні бази знань з всіма можливими правилами, включно з суперечливими правилами. Далі застосовують пряму та зворотні схеми жадібного алгоритму – за прямою схемою починають з порожньої бази знань і ітераційно додають по одному найкращому правилу. За зворотної схеми починають з повної бази знань, з якої на кожному кроці вилучають найгірше правило. Це дає змогу значно покращити точність та компактність бази знань.

Однією з переваг даного методу є можливість знехтувати експертом на етапі формування та відбору правил до нечіткої бази знань.

Також в ході проведення попередніх експериментів було встановлено, що пари суперечливих правил які потрапляють до найкращих нечітких баз знань є сусідні в частині консеквента, що є логічно. Тому можна покращити швидкодію жадібного адитивного алгоритму, зменшивши можливий простір вибору, скоротивши кількість кандидатів. А саме: при відборі якогось з правил, в наступних ітераціях розглядати лише сусідні, а інші відкидати, схематично алгоритм подано на рис. 2.3.

Для прикладу табл. 2.2, в нас відібралось правило: «якщо x_1 = Високий та x_2 = Низький то y = Середній», доцільно розглядати правила з консеквентами низький та високий, оскільки лише вони, можливо, покращать точність.

Таблиця 2.2 - Приклад відбору правила до нечіткої бази знань

Якщо		Тоді
x_1	x_2	y
Високий	Низький	Дуже низький
Високий	Низький	Низький
Високий	Низький	Середній
Високий	Низький	Високий
Високий	Низький	Дуже високий

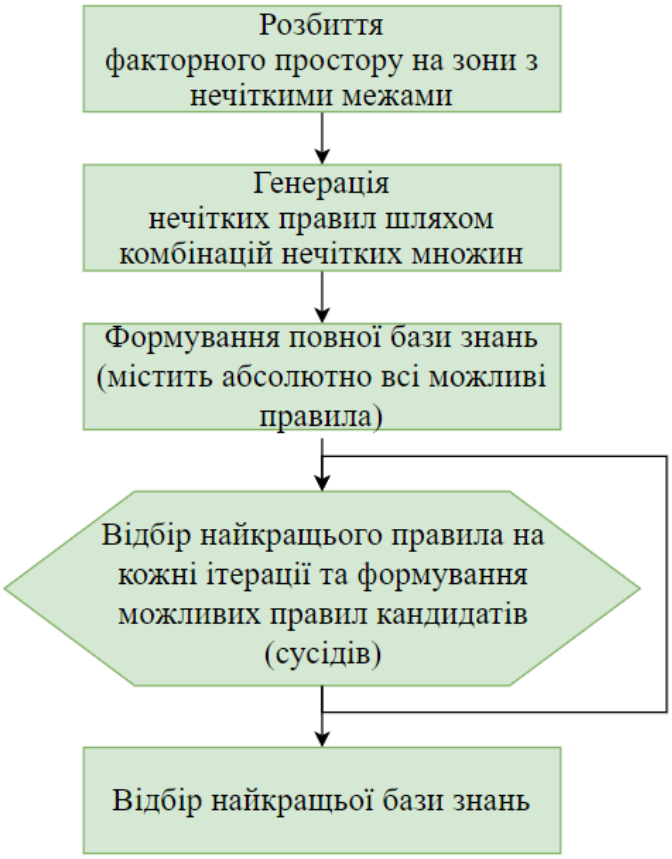


Рисунок 2.3 - Порядок дій алгоритму синтезу суперечливих нечітких баз знань з підмножини суперечливих, метод найближчого сусіда

Даний алгоритм, володіє рядом наступних переваг:

- значна швидкодія, адже тут не застосовуються ніякі перебори, а лише звичайна схема жадібного алгоритму, яку ми ще додатково скоротили;
- прогнозується покращення показника точності роботи отриманих баз знань, з суперечливими правилами, застосування яких підвищує «роздільну здатність» опису залежності, яку моделює база знань. Відповідно, з'являються шанси підвищити точність нечіткої бази знань;
- можливість знехтувати експертом на етапі синтезу та відбору правил, експерт потрібен лише для розбиття простору введення й виведення на нечіткі області.

Одним з недоліків даного алгоритму є те, що результуючі бази знань є неузгоджені, тобто мають суперечливі правила.

2.4 Синтез несуперечливих нечітких баз знань з підмножини суперечливих

Виникають ситуації коли потрібна максимальна інтерпретація бази знань, цього можна досягти шляхом синтезу баз знань без суперечливих правил, тобто, вона має бути узгоджена. Цього можна досягти шляхом модифікації алгоритму синтезу баз знань з суперечливими правилами шляхом накладання обмеження, а саме: при відборі якогось з правил відразу відкидати з такими ж антецедентами та відмінними консеквентами.

Як і в попередньому методі потрібно сформувати базу знань з всіма можливими правилами, включно з суперечливими правилами. Далі застосовують пряму схему жадібного алгоритму – ітераційно додаючи по одному найкращому правилу, і вводячи обмеження на правила з такими ж антецедентами, тобто, умовою якщо, наведено вище дії схематизовано на рис 2.4.

Для прикладу є база знань в якої лінгвістичне розбиття вхідних змінних здійснено із застосуванням 2-х термів вхідної змінної x_1, 3-х термів вхідної змінної x_2 та 2-х термів вихідної змінної y. Відповідно, максимальна кількість суперечливих нечітких правил (N_{max}) складає 2×3×2 = 12, табл. 2.3.

Таблиця 2.3 - Множини суперечливих правил-кандидатів

Якщо		Тоді
x_1	x_2	y
Низький	Низький	Низький
Низький	Низький	Середній
Низький	Низький	Високий
Низький	Високий	Низький
Низький	Високий	Середній
Низький	Високий	Високий
Високий	Низький	Низький
Високий	Низький	Низький
Високий	Низький	Середній
Високий	Низький	Високий
Високий	Високий	Низький
Високий	Високий	Середній

Наприклад: на перші ітерації в базу знань потрапило правило – «якщо x_1 = Низький та x_2 = Високий то у = Низький», тоді, інші правила з таким же антецедентами блокуються, і не будуть розглядатися в наступних ітераціях табл. 2.4.

Таблиця 2.4 – Правила- кандидати з однаковим антецедентами

Якщо		Тоді
x_1	x_2	у
Низький	Високий	Низький
Низький	Високий	Середній
Низький	Високий	Високий

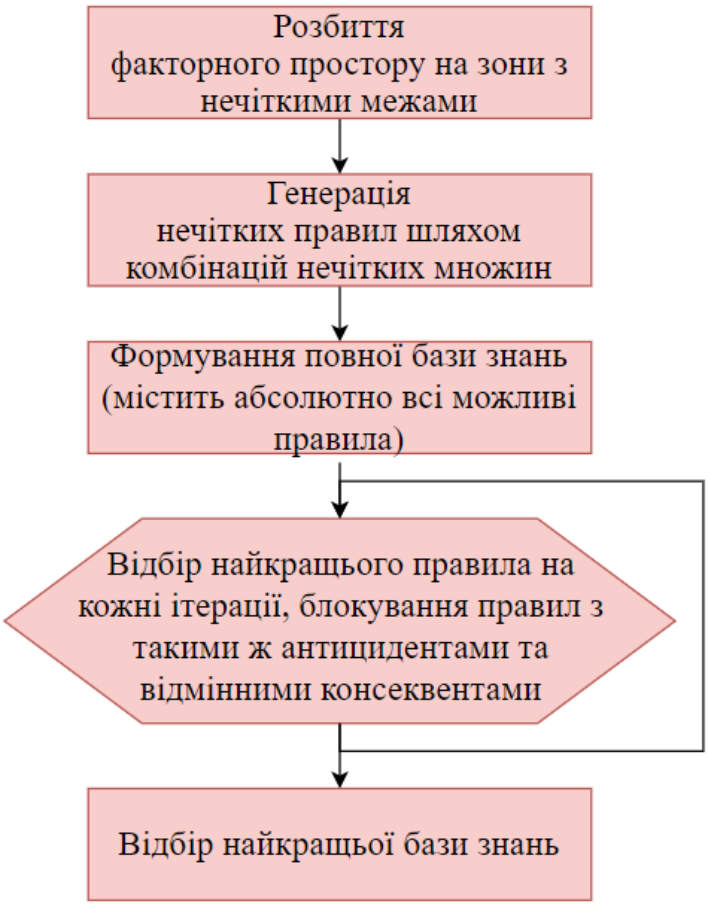


Рисунок 2.4 - Порядок дій алгоритму синтезу несуперечливих нечітких баз знань з підмножини суперечливих

Цей підхід також дозволяє відкинути експерта на етапі формування та відбору правил до нечіткої бази знань.

Основна відмінність даного методу полягає в тому, що в результаті ми отримуємо несуперечливу базу знань. Також процес відбору правил триває значно швидше за рахунок відкидання правил на кожні ітерації, що скорочує простір для вибору правила. Але точність даного алгоритму не така висока як в попереднього, адже отримані бази знань без суперечливих правил.

Висновки до розділу 2

В даному розділі наведено власне бачення алгоритмів синтезу та відбору правил нечітких баз знань та на основі розглянутих підходів запропоновані власні алгоритми, з використанням суперечливих правил. Точність яких, як передбачається, буде кращою, за наявні алгоритми. До того ж, запропоновані алгоритми не є складними, і тому, складні бази знань синтезуються дуже швидко.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

В даному розділі наведено як здійснюється нечітка ідентифікація в системі Matlab. Продемонстрована архітектура системи, які модулі наявні в системі та які функції вони виконують. Коротко описано процес розробки.

3.1 Нечітка ідентифікація в системі Matlab

Для розгляду результатів розробки та функціонування системи нечіткого висновку використовуватимемо графічні засоби пакету Fuzzy Logic Toolbox [40, 41, 42, 43]. Ці засоби використовуються і при розробці систем нечіткого висновку як графічна об'єктно-орієнтована мова автоматичного програмування.

До складу цих пакету входять:

- редактор систем нечіткого виведення *FIS Editor* (FIS);
- редактор функцій належності систем нечіткого виведення *Membership Function Editor* (MFE);
- редактор правил систем нечіткого висновку *Rule Editor*;
- програма перегляду правил системи нечіткого виведення *Rule Viewer*;
- програма перегляду поверхні нечіткого виведення *Surface Viewer*.

Говорячи про нечітку логіку, найчастіше мають на увазі системи нечіткого висновку, які лежать в основі різних експертних та керуючих процесах. Основними етапами нечіткого висновку є:

- формування основи правил системи нечіткого висновку;
- фазифікація вхідних параметрів;
- агрегування;
- активізація умов у нечітких правилах продукції;
- дефазифікація.

Ця схема відноситься до алгоритму нечіткого висновку Мамдані, який один з перших знайшов застосування в системах нечітких множин.

Редактор FIS є основним засобом, який використовується для створення та редагування систем нечіткого висновку у графічному режимі. Виклик редактора FIS для створення таких систем здійснюється за допомогою введення функції fuzzy у вікні команд системи MATLAB. Виклик редактора [44] вже створеної систем нечіткого висновку здійснюється введенням цієї функції, але з аргументом - ім'ям файлу, що її містить.

Редактор дає можливість описати основні властивості систем нечіткого висновку: тип та її структуру, способи реалізації операцій агрегування, активації та дефазифікації, а також описати лінгвістичні змінні, які використовуються при описі реальних моделей.

Опис виконується за допомогою графічного інтерфейсу через вікно редактора. Інтерфейс також дозволяє викликати інші редактори та програми перегляду.

На рис. 3.1, в верхні частині вікна редактора FIS, наведена схема системи нечіткого висновку, встановлена за замовчуванням. Лівий та правий прямокутники – вхідна та вихідна лінгвістичні змінні, відповідно, названі за замовчуванням. Центральний прямокутник відображає процесор нечітких правил [45](названий за замовченням Untitled). Сукупність цих правил описується в базі знань і, власне, визначає функціонування системи нечіткого висновку.

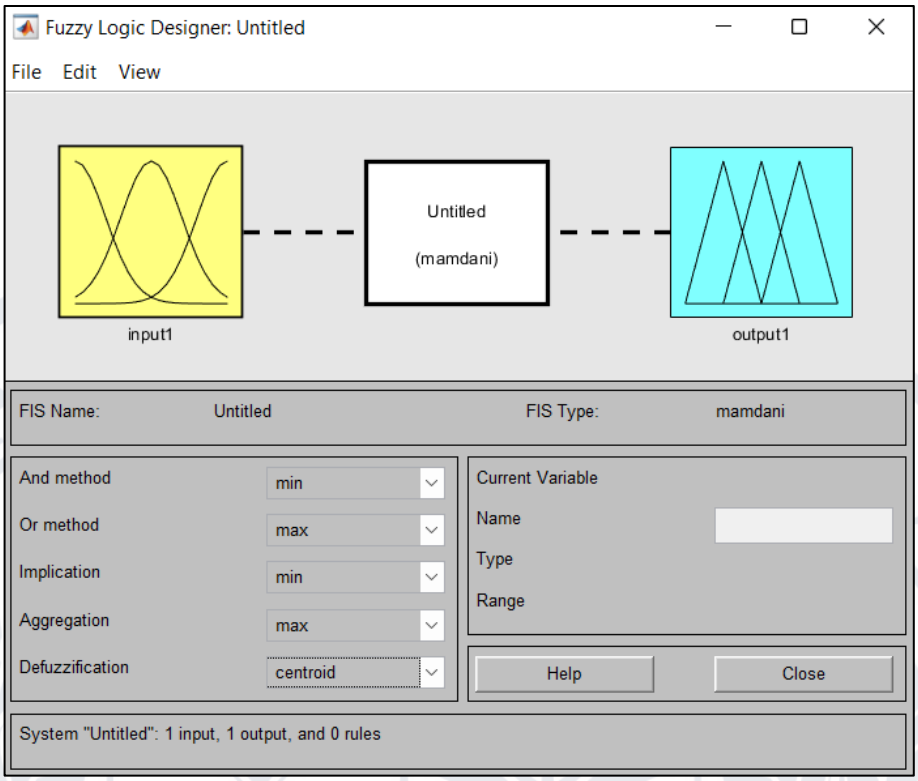


Рисунок 3.1 – Схема системи нечіткого висновку в вікні редактора FIS

За замовченням також задається цілий ряд параметрів: тип системи нечіткого висновку, нечіткі логічні операції, методи імплікації, агрегування та дефазифікації.

Зміна цих параметрів може виконуватись наступним чином[46, 47]:

Для додавання вхідної змінної потрібно виконати команду основного меню Edit > AddVariable… > Input. Після цього кількість лівих прямокутників збільшиться на один, з ім’ям за замовченням «input».

Для видалення вхідної змінної слід виділити її натиснувши на її зображення та натиснути клавішу Delete.

Для методів виконання операцій нечіткого висновку зміни проводяться вибором потрібних пунктів з п’яти випадаючих списків меню в лівій нижній частині вікна.

Для метода з’єднання виразів кон’юнкцією (меню And method) вибором елемента меню або min, тобто min-кон’юнкції, або prod, тобто метода алгебраїчного добутку ступені істинності поєднання нечітких виразів.

Для метода з’єднання виразів диз’юнкцією (меню Or method) вибором елемента меню або max, тобто max-диз’юнкції, або probor, тобто метода алгебраїчної суми ступенів істинності поєднуваних нечітких виразів.

Метод виводу висновку (меню Implication) може бути встановлений вибором елемента меню або min, тобто min-активації, або prod, тобто prod-активації.

Метод для агрегації значень функції належності кожної з вхідних лінгвістичних змінних висновків нечітких правил (меню Aggregation) може бути встановлений вибором елементів меню або max, тобто max-диз’юнкції, або sum, тобто метод граничної суми, або probor, метод алгебраїчної суми для об’єднуваних значень змінних.

Метод для виконання дефазифікації (меню Deffazification) може бути встановлений вибором елементів або centroid, тобто метод центра тяжіння для дискретної множини значень функцій належності, або bisector, тобто метод центру площі, або mom, метод середнього максимуму, або som, метод найменшого (лівого) модального значення, або lom, метод найбільшого (правого) модального значення.

Всі параметри наведені вище, в експериментальних дослідженнях цієї роботи обрані за замовченням.

Редактор функцій належності рис. 3.2 в графічному режимі забезпечує задання та зміну функцій належності будь-яких термів лінгвістичних змінних систем нечіткого висновку.

Для фазифікації лінгвістичної змінної системи нечіткого висновку слід виділити її зображення – названий прямокутник в лівій частині вікна редактора.

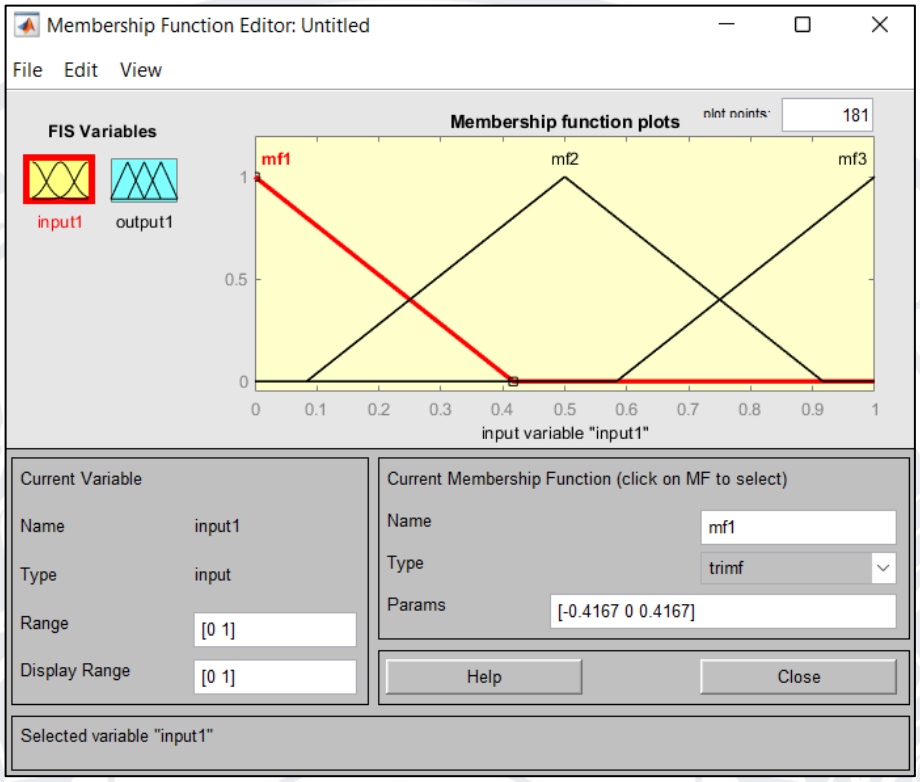


Рисунок 3.2 – Редактор функцій належності (MFE)

В вікні редактора виводяться графіки функцій належності для всіх значень обраної лінгвістичної змінної (за замовченням для трьох значень).

Для опису функцій належності кожного значення лінгвістичної змінної використовується 3 поля [48]: Name, Type і Params. Описувана функція виділяється кліком по її графіку. В полі Name встановлюється значення лінгвістичної змінної. В полі Type, вибором елемента меню, встановлюється імя потрібної функції належності (однієї з 11-ти вбудованих). В поле вводу Params вказується необхідні параметри функції належності, які визначають положення її модальних значень на числові шкалі, діапазон зміни якої вказується в полях вводу Range та Display range.

Ці операції виконуються над всіма значеннями із терм-множин лінгвістичних змінних системи нечіткого висновку.

Додавання нового значення лінгвістичної змінної з вбудованою функцією належності виконується по команді основного меню Edit > Add MF

Видалення непотрібного значення лінгвістичної змінної виконується натиском клавіші Delete, після виділення графіка функції належності цього значення.

Редактор правил рис. 3.3 системи нечіткого висновку забезпечує опис правил в графічному режимі. Основою мови представлення знань є апріорні елементарні нечіткі висловлювання відносно значень лінгвістичної змінної виду [49]:

<Ім’я лінгвістичної змінної is значення лінгвістичної змінної >.

Складні нечіткі висловлювання в умовних частинах нечітких конструкцій поєднуються зв’язками «and» і / або «or».

Умовна частина нечіткого правила вводиться словом «if» і розділяється заключним словом «then».

Після заключної частини в правилі в дужках вказується значення вагового коефіцієнта правила (поле вводу Weight).

Таким чином база знань системи нечіткого висновку, описана засобами пакету Fuzzy Logic ToolBox, представляється лінійною послідовністю нумерованих нечітких продукцій, описаних наведеним вище чином

Сформовані правила розміщуються на полі в верхній частині вікна редактора правил.

Опис нових або зміна нечітких конструкцій виконується тільки після опису всіх необхідних лінгвістичних змінних засобами редакторів FIS і MFE. Після цього можуть виконуватись 3 види операцій над правилами: видалення

(кнопка Delete rule), додавання (кнопка Add rule) та редагування (кнопка Change rule).

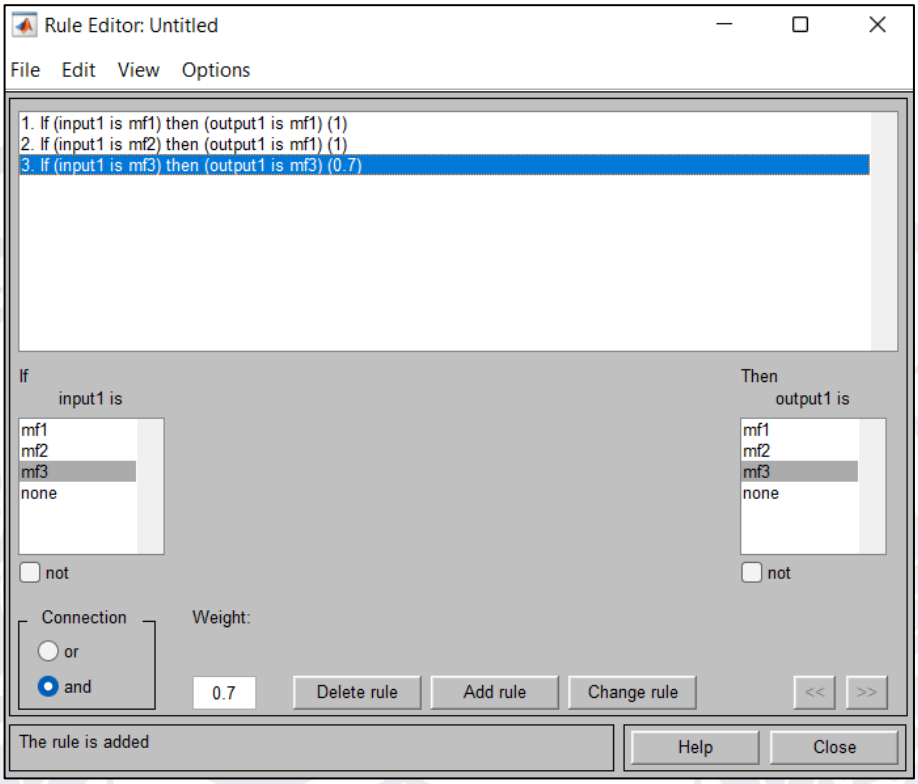


Рисунок 3.3 – Вікно редактора правил (Rule Editor)

Правило яке видаляється чи змінюється попередньо має бути виділено. Процес зміни або додавання правила потрібних значень лінгвістичної змінної, для опису апріорних висловлювань, виконується виділянням їх з списків, розміщених в полях нижньої частини вікна редактора. Тип зв'язки для складних конструкцій встановлюється перемикачем Connection. Величина вагового коефіцієнта правила встановлюється в полі вводу Weight.

Програма перегляду правил рис. 3.4 забезпечує ввід початкових даних та представлення отриманих результатів.

Значення лінгвістичної змінної, використаної в правилах, відображається в вікні програми в вигляді прямокутників з графіком їх функцій належності.

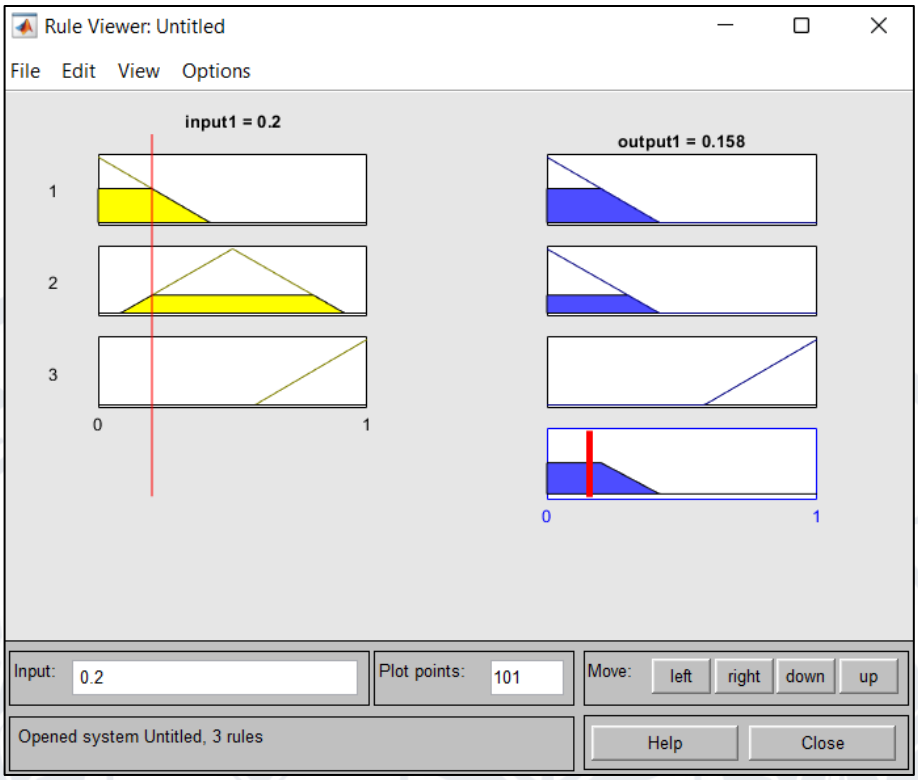


Рисунок 3.4 – Вікно програми перегляду правил (Rule Viewer)

Прямокутники, які стосуються однієї лінгвістичної змінної, розміщуються в стовпці з її ім'ям.

Число прямокутників у стовпці визначається числом правил у базі знань. Стовпець вихідної змінної доповнюється прямокутником з результатом дефазифікації вибраним методом.

Вертикальна лінія, що перетинає прямокутники стовпця вхідної змінної, показує введене значення числового еквівалента значення нечіткої лінгвістичної змінної, яке визначається завданням діапазону числової шкали («Range») та його функції належності на ній. Результат зчитується вгорі стовпця вихідної лінгвістичної змінної.

Для оцінки взаємовпливу значень лінгвістичної змінної на результати роботи системи нечіткого висновку можна скористатися **програмою перегляду поверхні** нечіткого виводу системи рис. 3.5.

Виклик програми забезпечується з вікна будь-якої розглянутої програми вибором меню вікна пункту View зі значенням Surface.

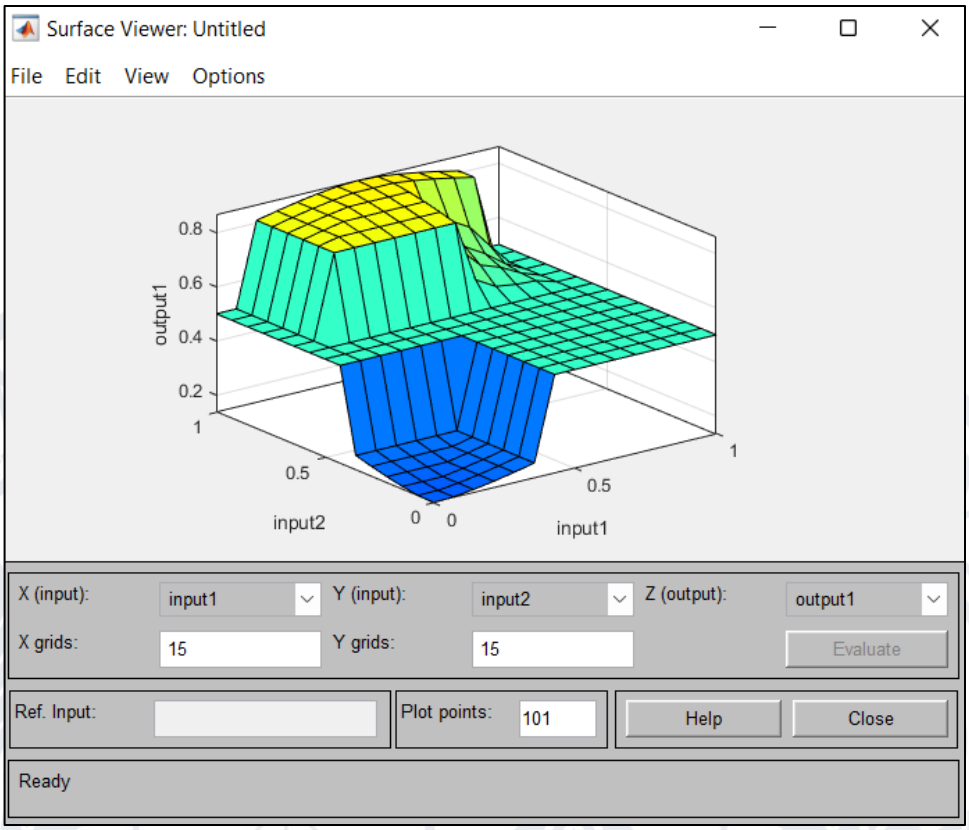


Рисунок 3.5 - Програма перегляду поверхні нечіткого виводу системи

3.2 Архітектура системи

Всі модулі які використовуються в дані роботі розроблені власноруч. Реалізовано модулі адитивних та субтрактивних алгоритмів, також модулі синтезу правил класичним алгоритмом Ванга-Менделя та його модифікацій: «сума голосів» та «головний конкурент», для зручної роботи додатково було розроблено модуль для візуалізації кривих навчання та поверхонь отриманих баз знань

Архітектура системи подана на рис. 3.6.

Рисунок 3.6 – Архітектура системи

3.3 Модуль реалізації адитивних та субтрактивних алгоритмів відбору правил нечітких баз знань

Дані модулі призначені для адитивних та субтрактивних алгоритмів відбору правил нечітких баз знань.

На вхід вони приймають:

- повну базу знань з якої потрібно відібрати найкращі правила;
- дані в форматі вхід-вихід, для оцінки якості правил та їх впливу при додаванні в базу знань;
- та додаткові параметри які можна було б обчислити в цьому модулі, але з оптимізаційної точки зору вони подаються як аргументи, це: кількість правил бази знань без суперечливих правил та з суперечливими.

Модуль повертає:

- масив даних мінімального значення RMSE для баз знань різної розмірності від розмірності з 1 правилом до максимальної розмірності;

- індекс та мінімальне значення RMSE найкращої бази знань, власне це і є те, що ми шукаємо.

Алгоритм функціонує наступним чином:

Для прямої схеми (адитивний), з бази знань викидаються всі правила. Організовується цикл в якому на кожному кроці вибирається правило яке приводить до найбільшого зменшення середньоквадратичної нев'язкий, з урахуванням доданих до цього правил. З усіх ітерацій вибирається крок на якому значення RMSE було мінімальним, власне на цьому кроці й була сформована найкраща база знань, яку нам поверне функція.

Для зворотної схеми (субтрактивний), організовується цикл в якому на кожному кроці відкидається найгірше правило, тобто додавання якого приводило до найбільшого збільшення середньоквадратичної нев'язкий, з урахуванням відкинутих до цього правил. З усіх ітерацій вибирається крок на якому значення RMSE було мінімальним, власне на цьому кроці й була сформована найкраща база знань, яку нам поверне функція.

Детальний лістинг наведено в додатку Б, для адаптивного алгоритму та в додатку В для субтрактивного.

Результат роботи подано на рис. 4.13-4.16, 4.20-4.23, 4.24-4.27.

Також в ході проведення попередніх експериментів було встановлено, що пари суперечливих правил які потрапляють до найкращих нечітких баз знань є сусідні в частині консеквента, що є логічно. Тому можна покращити швидкодію жадібного адитивного алгоритму, зменшивши можливий простір вибору, скоротивши кількість кандидатів. Тобто, при відборі якогось з правил, в наступних ітераціях розглядати лише сусідні, а інші відкидати, дану модифікацію подано в додатку Г.

Також було проведено модифікацію додаток Д для отримання баз знань без суперечливих правил шляхом відбору з суперечливих баз знань. Адже може виникнути ситуація коли потрібна максимальна інтерпретація бази знань, і щоб отримана база знань була узгоджена.

3.4 Модуль реалізації класичного алгоритму Ванга-Менделя та запропонованих модифікацій

Дані модулі призначені синтезу правил нечітких баз знань.

На вхід вони приймають:

порожню базу знань (без правил), для отримання даних стосовно розбиття факторного простору на нечіткі області;

дані в форматі вхід-вихід для синтезу правил.

На виході ми маємо бази знань з сформованими лінгвістичними правилами.

Алгоритм функціонування подано в розділі 1.4 та на рис. 2.1, 2.2.

Лістинг для класичного алгоритму Ванга-Менделя подано в додатку Е, для модифікації алгоритму Ванга-Менделя, а саме «сума голосів» подано в додатку Ж, для модифікації алгоритму Ванга-Менделя, а саме «головний конкурент» подано в додатку К.

Результат роботи подано нижче на рис. 3.7, 3.8, 3.9, для опису залежності використано 4, 3 і 5 термів для змінних x_1, x_2 та y, відповідно. Максимальна кількість несуперечливих нечітких правил складає 4×3 =12, а максимальна кількість правил алгоритму «головний конкурент» складає 12×2 = 24.

```
1     "x_1==Низький & x_2==Низький => y=ВищеСереднього (1)"
2     "x_1==Низький & x_2==Середній => y=Середній (1)"
3     "x_1==Низький & x_2==Високий => y=Середній (1)"
4     "x_1==Середній & x_2==Низький => y=Низький (1)"
5     "x_1==Середній & x_2==Середній => y=ДужеНизький (1)"
6     "x_1==Середній & x_2==Високий => y=ДужеНизький (1)"
7     "x_1==Високий & x_2==Низький => y=Низький (1)"
8     "x_1==Високий & x_2==Середній => y=Низький (1)"
9     "x_1==Високий & x_2==Високий => y=Низький (1)"
10    "x_1==ВищеСереднього & x_2==Низький => y=Низький (1)"
11    "x_1==ВищеСереднього & x_2==Середній => y=Низький (1)"
12    "x_1==ВищеСереднього & x_2==Високий => y=Низький (1)"
```

Рисунок 3.7 – Правила синтезовані класичним алгоритмом Ванга-Менделя

```
1     "x_1==Низький & x_2==Низький => y=ДужеВисокий (1)"
2     "x_1==Низький & x_2==Середній => y=ВищеСереднього (1)"
3     "x_1==Низький & x_2==Високий => y=Середній (1)"
4     "x_1==Середній & x_2==Низький => y=ДужеНизький (1)"
5     "x_1==Середній & x_2==Середній => y=ДужеНизький (1)"
6     "x_1==Середній & x_2==Високий => y=ДужеНизький (1)"
7     "x_1==Високий & x_2==Низький => y=ДужеНизький (1)"
8     "x_1==Високий & x_2==Середній => y=ДужеНизький (1)"
9     "x_1==Високий & x_2==Високий => y=ДужеНизький (1)"
10    "x_1==ВищеСереднього & x_2==Низький => y=Низький (1)"
11    "x_1==ВищеСереднього & x_2==Середній => y=Низький (1)"
12    "x_1==ВищеСереднього & x_2==Високий => y=Низький (1)"
```

Рисунок 3.8 – Правила синтезовані модифікованим алгоритмом Ванга-Менделя «сума голосів»

```
1     "x_1==Низький & x_2==Низький => y=ВищеСереднього (1)"
2     "x_1==Низький & x_2==Середній => y=Середній (1)"
3     "x_1==Низький & x_2==Середній => y=ВищеСереднього (1)"
4     "x_1==Низький & x_2==Високий => y=Середній (1)"
5     "x_1==Середній & x_2==Низький => y=Низький (1)"
6     "x_1==Середній & x_2==Середній => y=ДужеНизький (1)"
7     "x_1==Середній & x_2==Середній => y=Низький (1)"
8     "x_1==Середній & x_2==Високий => y=ДужеНизький (1)"
9     "x_1==Високий & x_2==Низький => y=ДужеНизький (1)"
10    "x_1==Високий & x_2==Низький => y=Низький (1)"
11    "x_1==Високий & x_2==Середній => y=ДужеНизький (1)"
12    "x_1==Високий & x_2==Середній => y=Низький (1)"
13    "x_1==Високий & x_2==Високий => y=ДужеНизький (1)"
14    "x_1==Високий & x_2==Високий => y=Низький (1)"
15    "x_1==ВищеСереднього & x_2==Низький => y=ДужеНизький (1)"
16    "x_1==ВищеСереднього & x_2==Низький => y=Низький (1)"
17    "x_1==ВищеСереднього & x_2==Середній => y=ДужеНизький (1)"
18    "x_1==ВищеСереднього & x_2==Середній => y=Низький (1)"
19    "x_1==ВищеСереднього & x_2==Високий => y=ДужеНизький (1)"
20    "x_1==ВищеСереднього & x_2==Високий => y=Низький (1)"
```

Рисунок 3.9– Правила синтезовані модифікованим алгоритмом Ванга-Менделя «головний конкурент»

3.5 Додаткові модулі

Для зручної робити було реалізовано додаткові модулі, для зручної візуалізації кривих навчання алгоритмів та поверхонь залежностей отриманих баз знань.

На вхід вони приймають назву методів, мінімальні значення RMSE на кожному кроці, індекс та точність найкращих баз знань, черговість потрапляння правил до бази знань.

Результат модуля це всі візуалізації наведені в 4 розділі тобто рис. 4.1-4.27.

Детальний лістинг подано в додатку Л.

Висновки до розділу 3

В даному розділі було продемонстровано як здійснюється нечітка ідентифікація в системі Matlab. Наведена архітектура системи, які модулі наявні в системі та які функції вони виконують. Коротко описано процес розробки.

РОЗДІЛ 4
ЕКСПЕРЕМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

В даному розділі наведені обчислювальні експерименти із синтезу нечітких баз знань Мамдані з суперечливими правилами, головною метою яких є порівняння кривих навчання нечітких баз з та без суперечливих правил, як результат, доведення чи спростування доречності застосування суперечливих правил в нечітких базах знань.

4.1 Методика проведення досліджень

Експерименти проведемо за методикою з робіт [11, 30]. Відберемо 3 еталоні двофакторні залежності рис. 4.2, 4.4, 4.6, 4.8:

$$y = x_1\sqrt{x_2},\ x_1 \in [2,22],\ x_2 \in [3,16], \quad (а)$$

$$y = \frac{x_1^2+x_2}{0.1e^{x_1+x_2}},\ x_1 \in [-5,5],\ x_2 \in [3,7], \quad (б)$$

$$y = x_1^2 - x_2^3 * tan(0.1 * x_1) \in [-20,25],\ x_2 \in [-1,1.3]. \quad (в)$$

Також проведемо експеримент з реальним набором даних Auto-MPG [50], в якому наведені дані 398 експериментів про залежність запасу ходу автомобіля на 1 галон пального (y) від маси автомобіля (x_1) та часу розгону до швидкості 60 миль на годину (x_2). Власне по цим даним і сформуємо залежність y = f(x_1, x_2) (г). Для зручності використання було здійснено перехід до метричної системи, тобто, запас ходу з миль на галон було переведено до кілометрів на 1 літер пального; маса автомобіля з фунтів в кілограми, час розгону до 100 км/год. В цій залежності фактори впливу приймають наступні значення $x_1 \in$[731,7; 2331,5] та $x_2 \in$[8,2849 25,6833].

Для кожної з цих залежностей синтезуємо множину нечітких правил Мамдані. З цієї множини правил кандидатів сформуємо нечіткі бази знань різного обсягу і побудуємо криві навчання – залежності точності від кількості правил. Порівнюючи криві навчання на основі суперечливих та несуперечливих множин правил-кандидатів, зробимо висновок про доцільність використання суперечливих правил для нечіткої бази знань

Мамдані.

Для кожної з наведених залежностей експерименти проводяться за такою схемою :

1) спостерігаючи за тривимірним графіком аналітичної залежності сформуємо повну базу знань несуперечливих нечітких правил;

2) доповнимо вище сформовану базу знань Мамдані суперечливими правилами;

3) згенеруємо тестову вибірку з 100 точок з використанням аналітичних залежностей (а) – (в), в якості тестових значень для залежності (г), буде розглядатись власне сам датасет.

4) побудуємо криві навчання у формі залежності *RMSE* від кількості правил (*N*), використовуючи жадібний алгоритм відбору правил нечіткої бази знань. Експерименти проведемо для $N=\overline{1,N_{max}}$, де N_{max} – максимальне число несуперечливих правил. За жадібним алгоритмом на кожній ітерації в базу знань додається правило, яке забезпечує найбільше зменшення нев'язки RMSE.

Чітке моделювання здійснено в середовищі MATLAB з використанням пакета Fuzzy Logic Toolbox. Нечітке розбиття діапазону вхідних змінних здійснено за допомогою гаусових функцій належності.

4.2 Формування баз знань

Для залежності (а) лінгвістичне розбиття вхідних змінних здійснено із застосуванням 4-х термів, а вихідної змінної – із застосуванням 5-ти термів, рис. 4.1 Відповідно, максимальна кількість несуперечливих нечітких правил (N_{max}) складає 4×4 =16, табл. 4.1. А максимальна кількість суперечливих нечітких правил (N_{max}) складає 4×4×5 = 80.

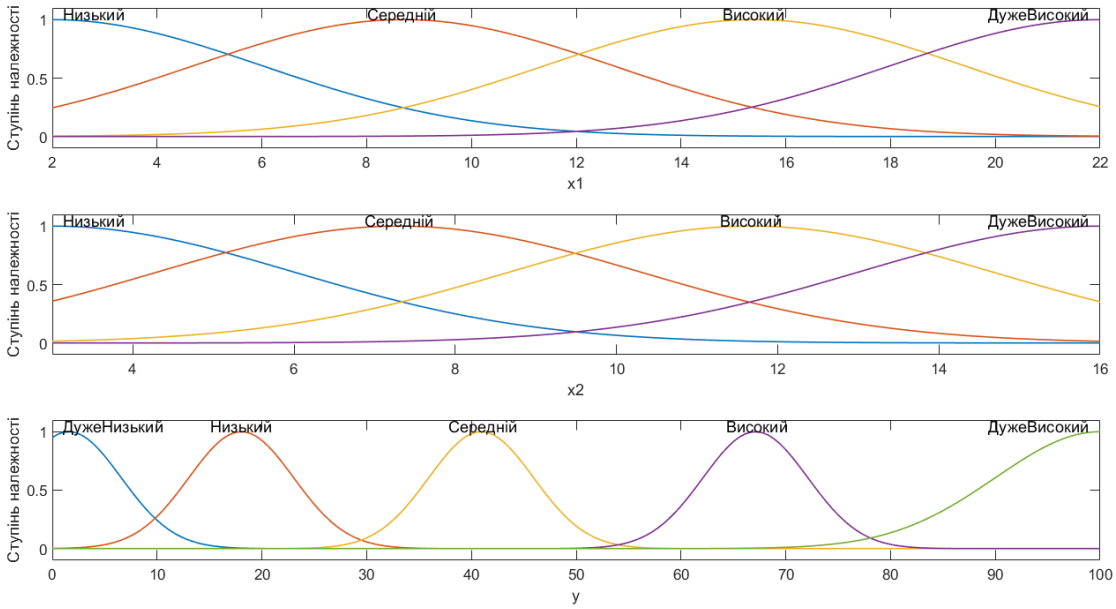


Рисунок 4.1 – Функції належності залежності (а)

Таблиця 4.1 – Множини несуперечливих правил-кандидатів для залежності (а)

Якщо		Тоді
x_1	x_2	y
Низький	Низький	Дуже низький
Низький	Середній	Дуже низький
Низький	Високий	Дуже низький
Низький	Дуже високий	Дуже низький
Середній	Низький	Низький
Середній	Середній	Низький
Середній	Високий	Низький
Середній	Дуже високий	Низький
Високий	Низький	Низький
Високий	Середній	Середній
Високий	Високий	Середній
Високий	Дуже високий	Середній
Дуже високий	Низький	Середній
Дуже високий	Середній	Середній
Дуже високий	Високий	Високий
Дуже високий	Дуже високий	Дуже високий

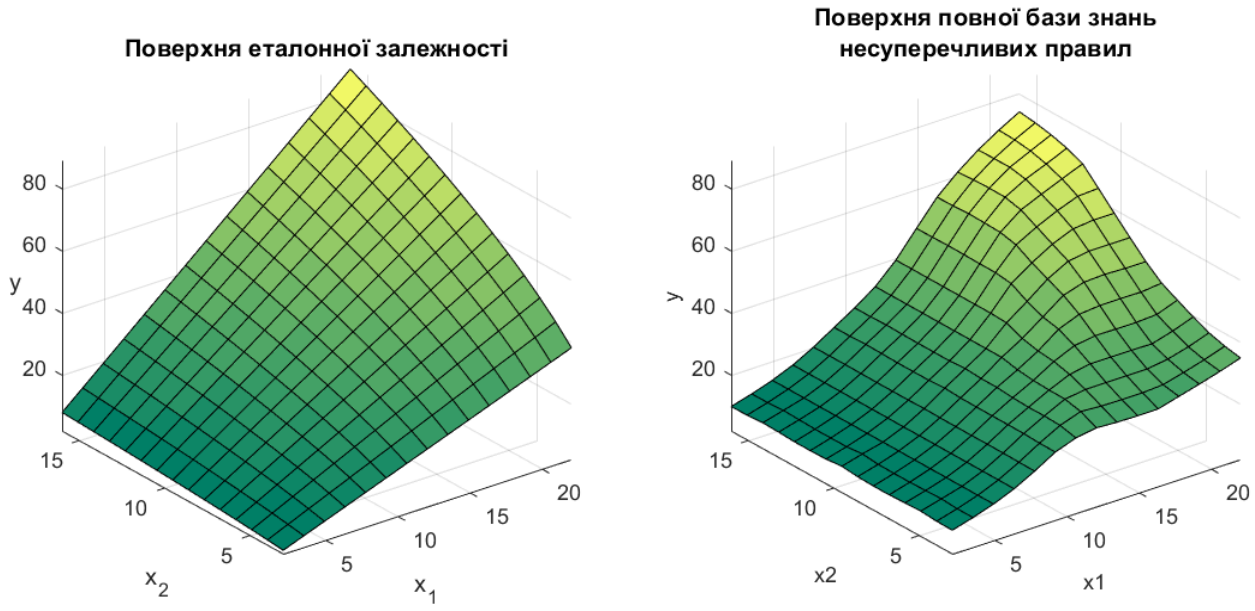


Рисунок 4.2 – Поверхня еталонної залежності (а) та поверхня сформованої повної бази знань без суперечливих правил

Для опису залежності (б) використано 4, 3 і 5 термів для змінних x_1, x_2 та y, відповідно, рис. 4.3. Максимальна кількість несуперечливих нечітких правил складає 4×3 =12, табл. 4.2, а суперечливих – 4×3×5 = 60.

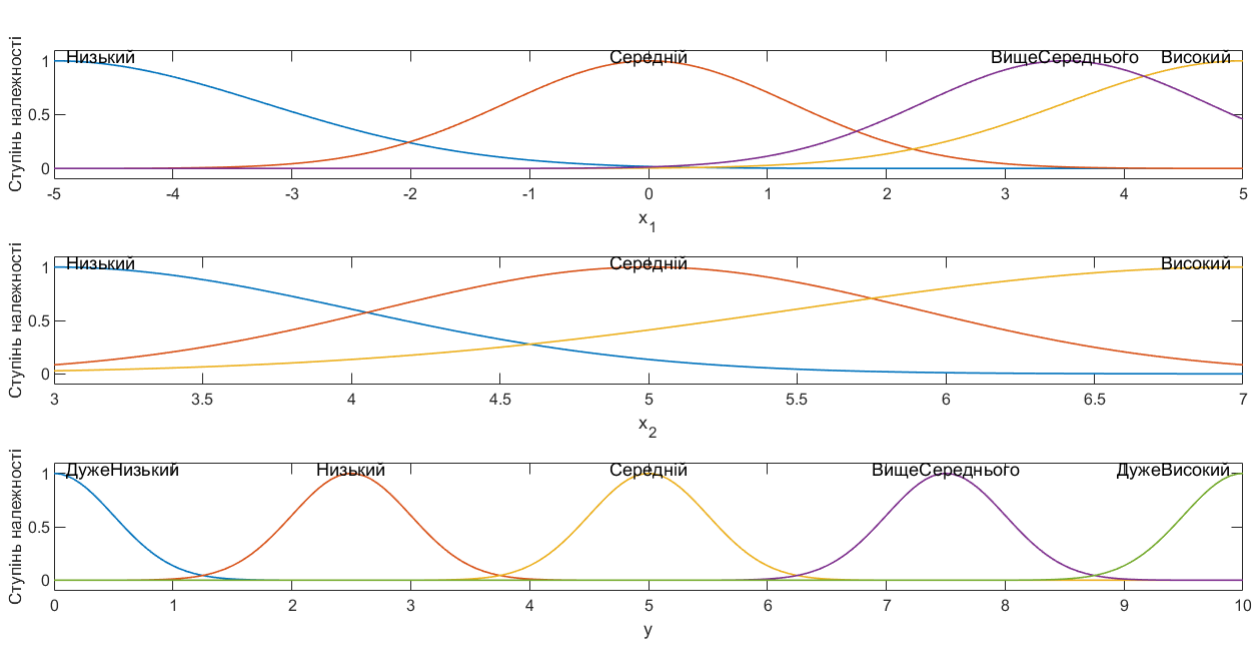


Рисунок 4.3 – Функції належності залежності (б)

Таблиця 4.2 – Множини несуперечливих правил-кандидатів для залежності (б)

Якщо		Тоді
x_1	x_2	y
Низький	Низький	Дуже високий
Низький	Середній	Вище середнього
Низький	Високий	Середній
Середній	Низький	Дуже низький
Середній	Середній	Дуже низький
Середній	Високий	Дуже низький
Високий	Низький	Дуже низький
Високий	Середній	Дуже низький
Високий	Високий	Дуже низький
Вище середнього	Низький	Низький
Вище середнього	Середній	Низький
Вище середнього	Високий	Низький

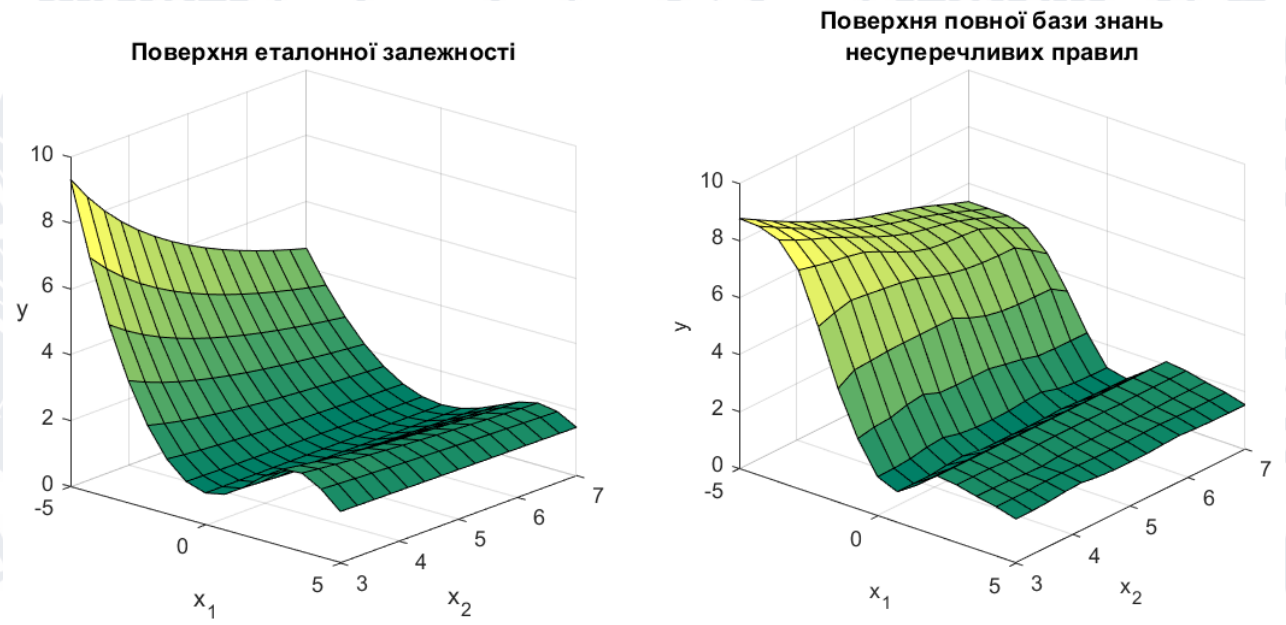


Рисунок 4.4 – Поверхня еталонної залежності (б) та поверхня сформованої повної бази знань без суперечливих правил

Для опису залежності (в) використано 4, 5 і 4 терми для змінних x_1, x_2 та y, відповідно, рис. 4.5. Максимальна кількість несуперечливих нечітких правил складає 4×5 =20, табл. 4.3, а суперечливих – 4×5×5 = 80.

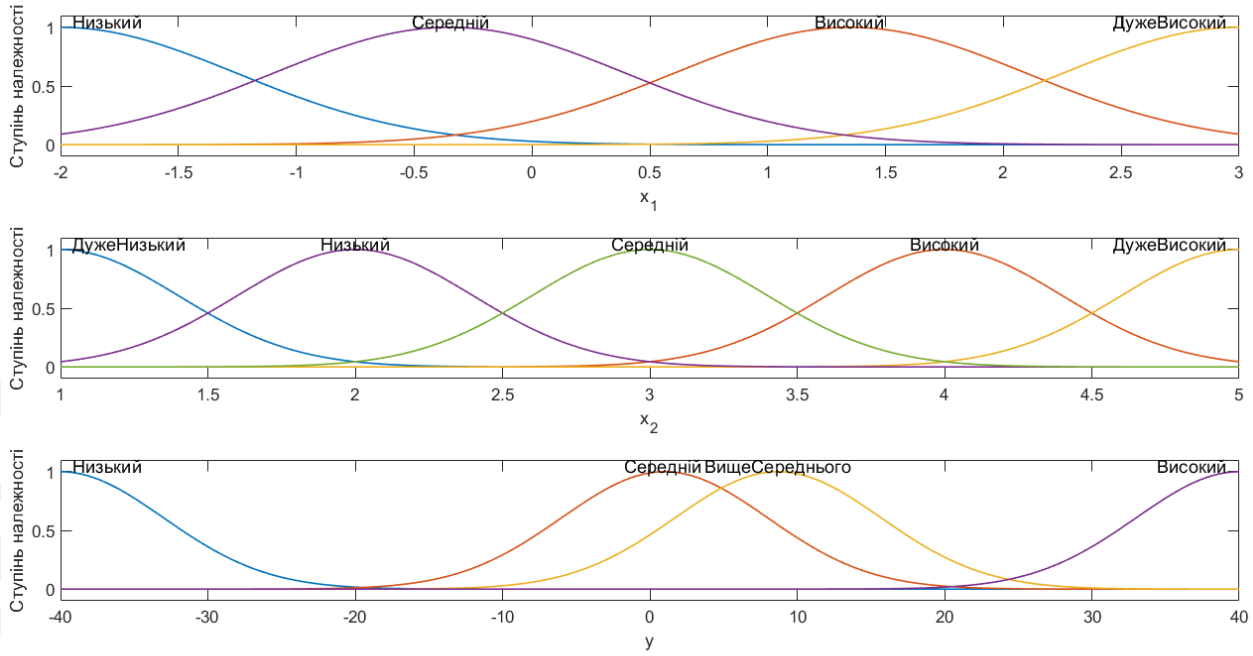


Рисунок 4.5 – Функції належності залежності (в)

Таблиця 4.3 – Множини несуперечливих правил-кандидатів для залежності (в)

Якщо		Тоді
x_1	x_2	y
Низький	Дуже низький	Середній
Низький	Низький	Вище середнього
Низький	Середній	Вище середнього
Низький	Високий	Вище середнього
Низький	Дуже високий	Високий
Середній	Дуже низький	Середній
Середній	Низький	Середній
Середній	Середній	Середній
Середній	Високий	Середній
Середній	Дуже високий	Середній
Високий	Дуже низький	Середній
Високий	Низький	Середній
Високий	Середній	Середній
Високий	Високий	Середній
Високий	Дуже високий	Низький
Дуже високий	Дуже низький	Вище середнього
Дуже високий	Низький	Вище середнього
Дуже високий	Середній	Середній
Дуже високий	Високий	Середній
Дуже високий	Дуже високий	Низький

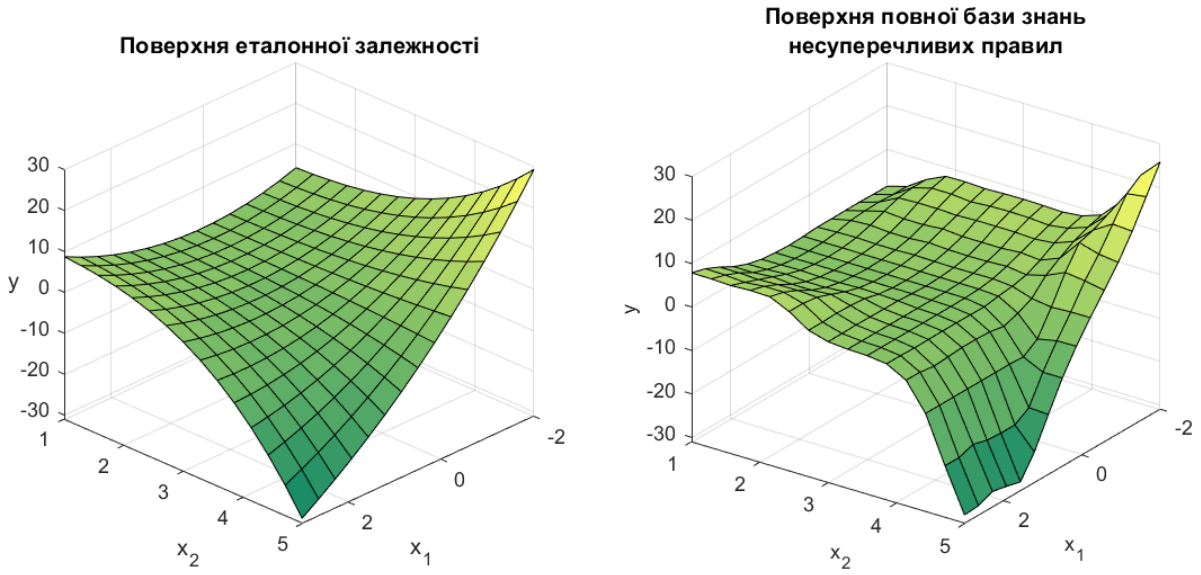


Рисунок 4.6 – Поверхня еталонної залежності (в) та поверхня сформованої повної бази знань без суперечливих правил

Для опису залежності (г) використано 4, 3 і 4 терми для змінних x_1, x_2 та y, відповідно, рис. 4.7. Максимальна кількість несуперечливих нечітких правил складає 4×3 =12 табл. 4.4, а суперечливих – 4×3×4 = 48.

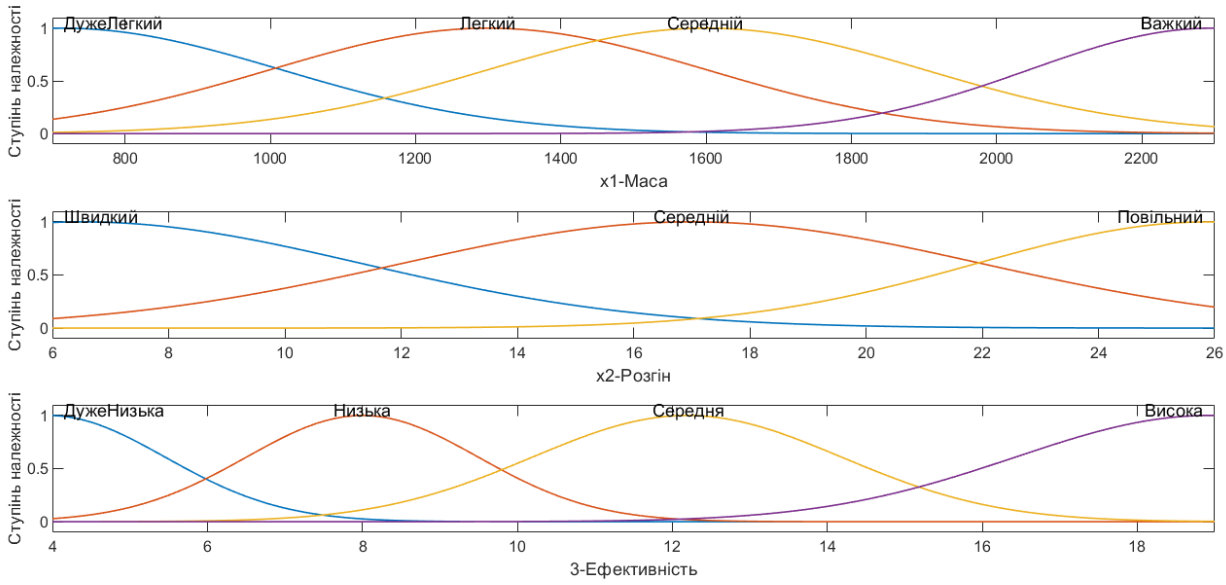


Рисунок 4.7 – Функції належності залежності (г)

Таблиця 4.4 – Множини несуперечливих правил-кандидатів для залежності (г)

Якщо		Тоді
x_1 - маса авто	x_2 - прискорення	y - паливна ефективність
Дуже легкий	Швидкий	Середня
Дуже легкий	Середній	Висока
Дуже легкий	Повільний	Висока
Легкий	Швидкий	Низька
Легкий	Середній	Середня
Легкий	Повільний	Середня
Середній	Швидкий	Низька
Середній	Середній	Низька
Середній	Повільний	Середня
Важкий	Швидкий	Дуже низька
Важкий	Середній	Дуже низька
Важкий	Повільний	Дуже низька

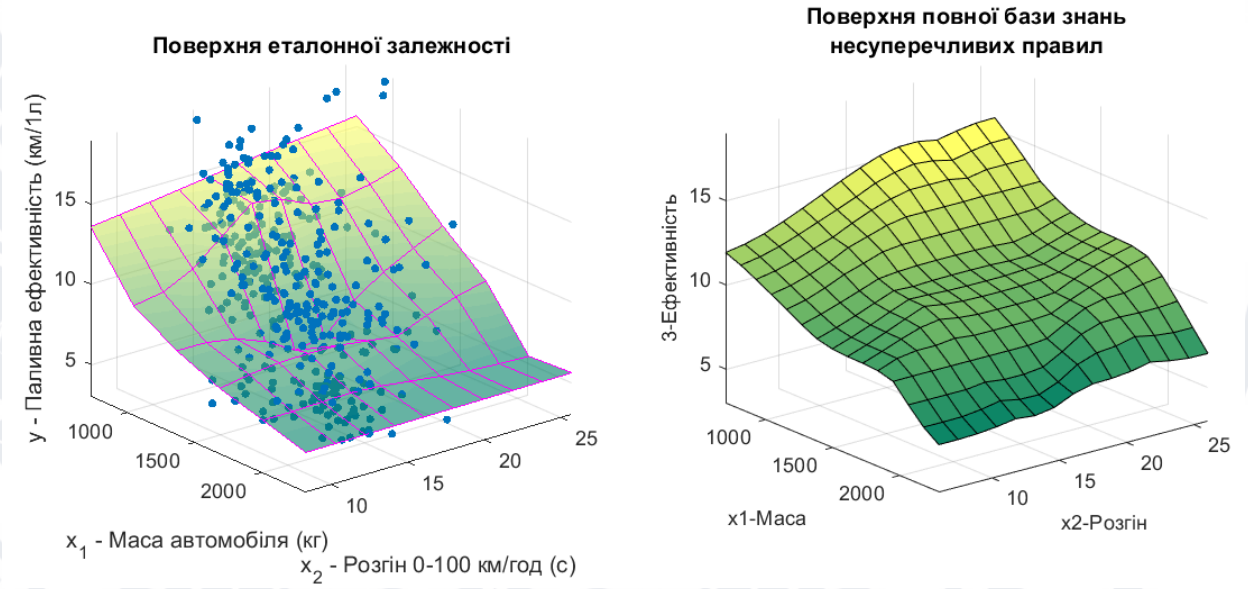


Рисунок 4.8 – Поверхня еталонної залежності (г) та поверхня сформованої повної бази знань без суперечливих правил

4.3 Генерація тестової вибірки

Для еталоних залежностей (а-в) сформовано сітку розмірності 10 × 10 по змінних x_1 та x_2 і далі ці значення були подані в функцію для отримання значення y, таким чином було отримані тестові значення рис. 4.9, 4.10, 4.11.

В якості тестових значень для залежності(г) виступає власне сам набір даних рис. 4.12.

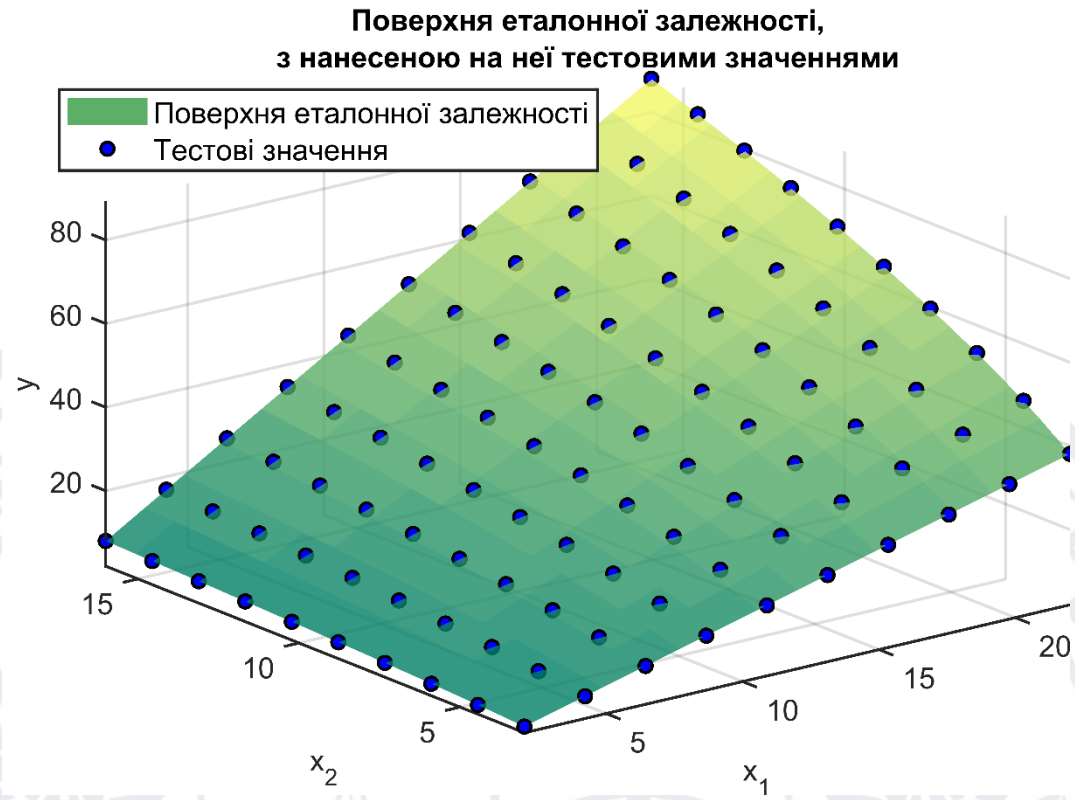


Рисунок 4.9 – Поверхня еталонної залежності (а) з нанесеними на неї тестовими значеннями

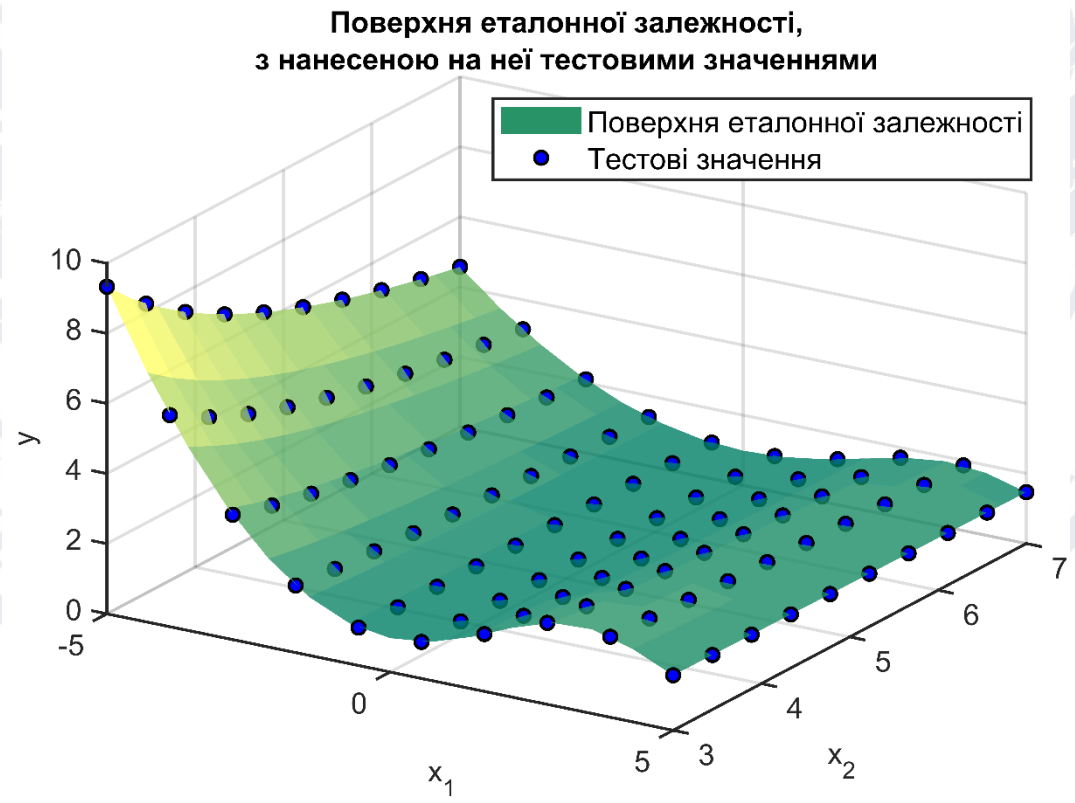


Рисунок 4.10 – Поверхня еталонної залежності (б) з нанесеними на неї тестовими значеннями

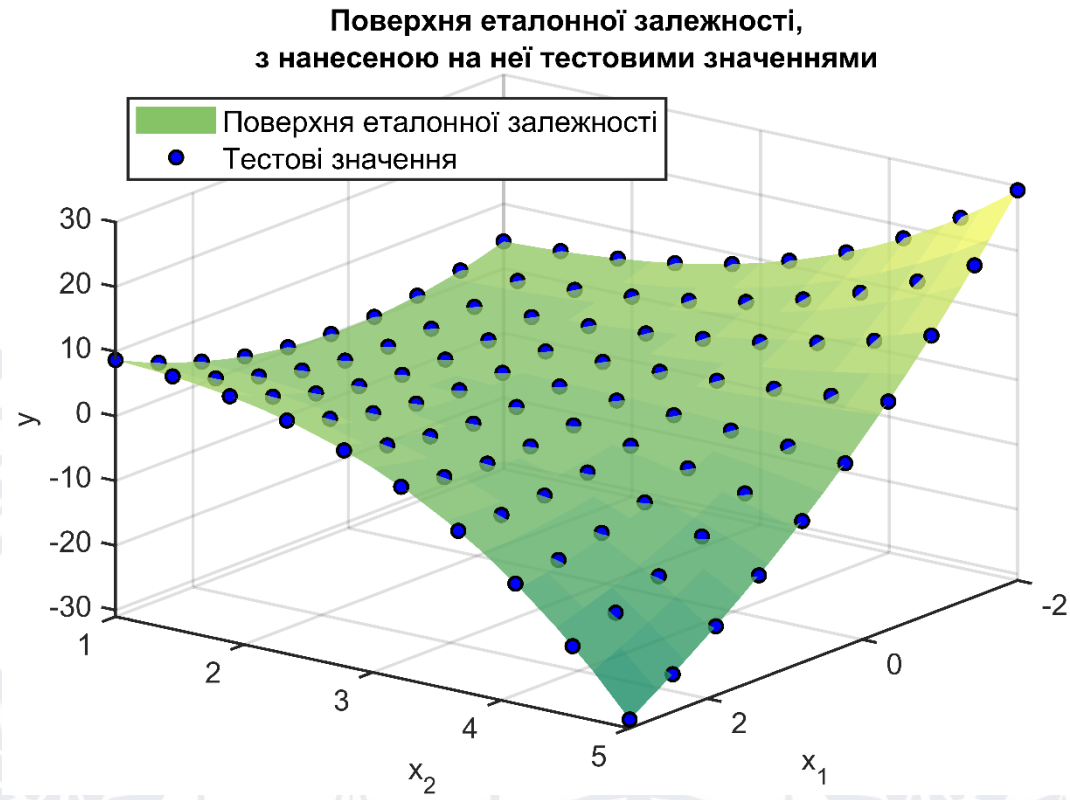


Рисунок 4.11 – Поверхня еталонної залежності (в) з нанесеними на неї тестовими значеннями

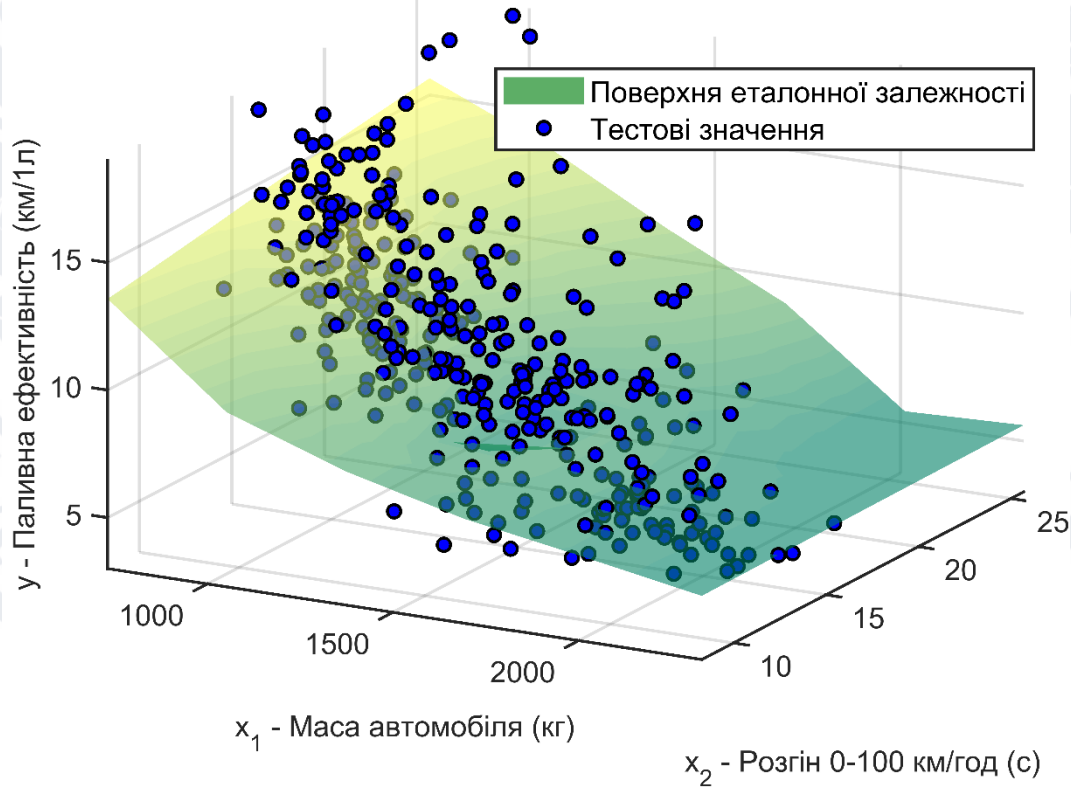


Рисунок 4.12 – Поверхня еталонної залежності (г) з нанесеними на неї тестовими значеннями

4.4 Синтез баз знань з підмножини суперечливих та несуперечливих правил

Проведемо синтез баз знань з підмножини суперечливих та несуперечливих правил, та побудуємо криві навчання залежності показника точності від кількості правил. Криві навчання подано на рис. 4.13, 4.14, 4.15, 4.16.

Для близької до лінійної залежності (а) покращення за рахунок використання суперечливих правил доволі значне, з 4.3 до 2.65, також візуально видно, що поверхня найкращої нечіткої бази знань дуже добре описує задану залежність див. рис. 4.13.

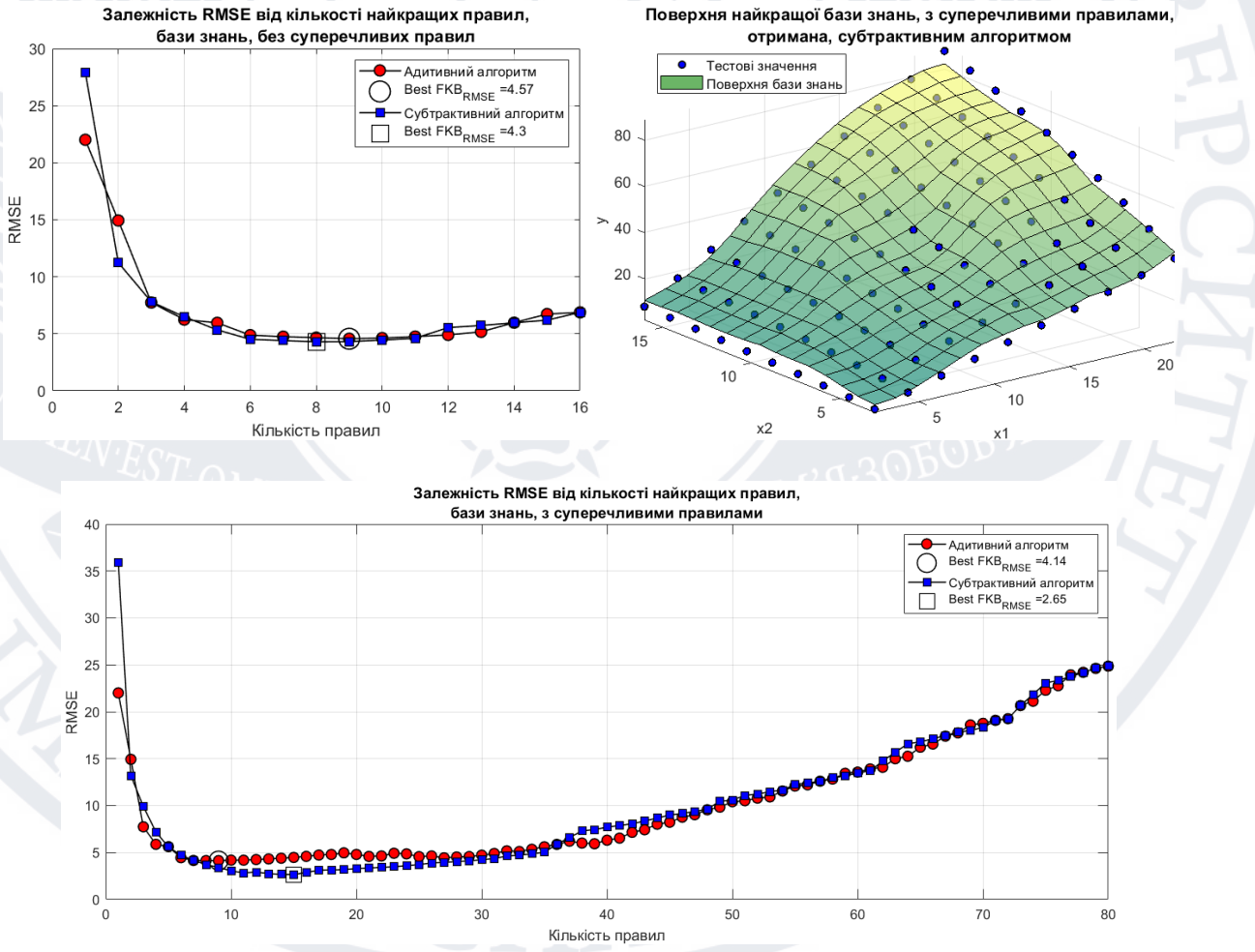


Рисунок 4.13 – Криві навчання баз знань з та без суперечливих правил залежності (а) , поверхня найкращої бази знань

Для суттєво нелінійної залежності (б) покращення за рахунок використання суперечливих не відбулося див. рис. 4.13.

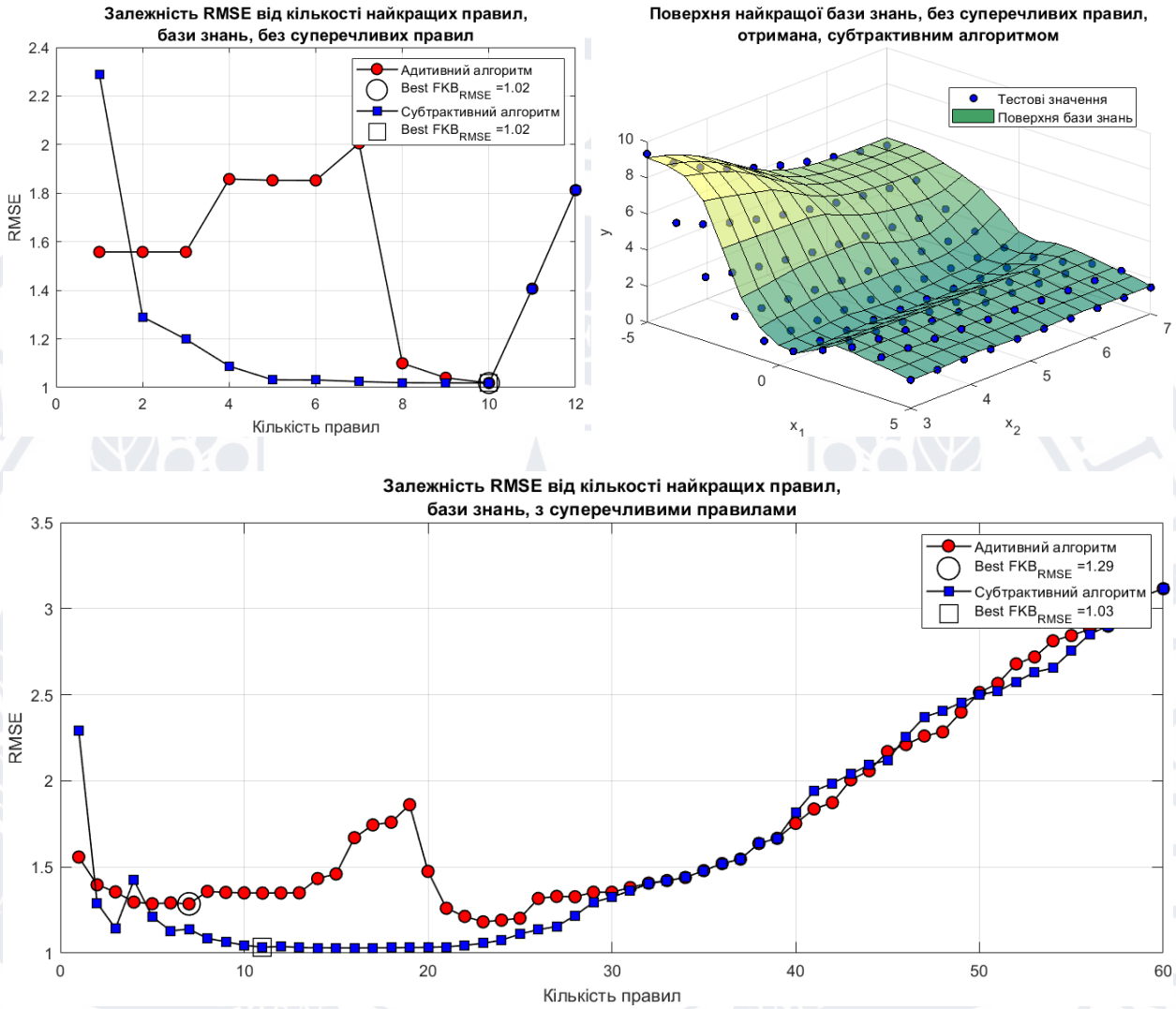


Рисунок 4.14 – Криві навчання баз знань з та без суперечливих правил залежності (б) , поверхня найкращої бази знань

А от для суттєво нелінійної залежності (в) покращення за рахунок використання суперечливих значне, з 3.29 до 2.41, також візуально видно, що поверхня найкращої нечіткої бази майже ідеально описує задану залежність див. рис. 4.15.

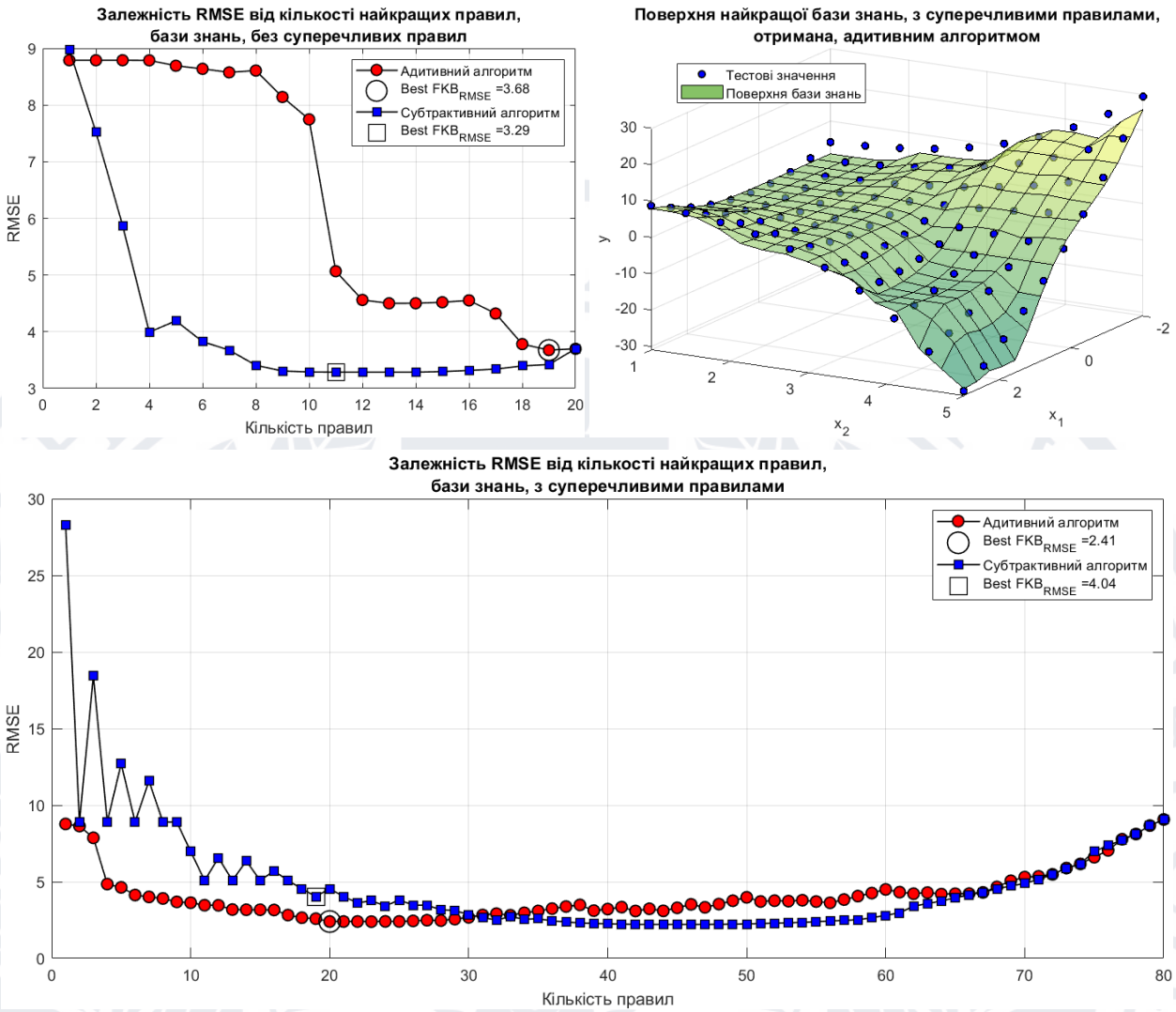


Рисунок 4.15 – Криві навчання баз знань з та без суперечливих правил залежності (в) , поверхня найкращої бази знань

Для близької до лінійної залежності (г) покращення за рахунок використання суперечливих не значне, з 2.05 до 1.8, візуально видно, що поверхня найкращої нечіткої бази знань не погано описує задану залежність див. рис. 4.16.

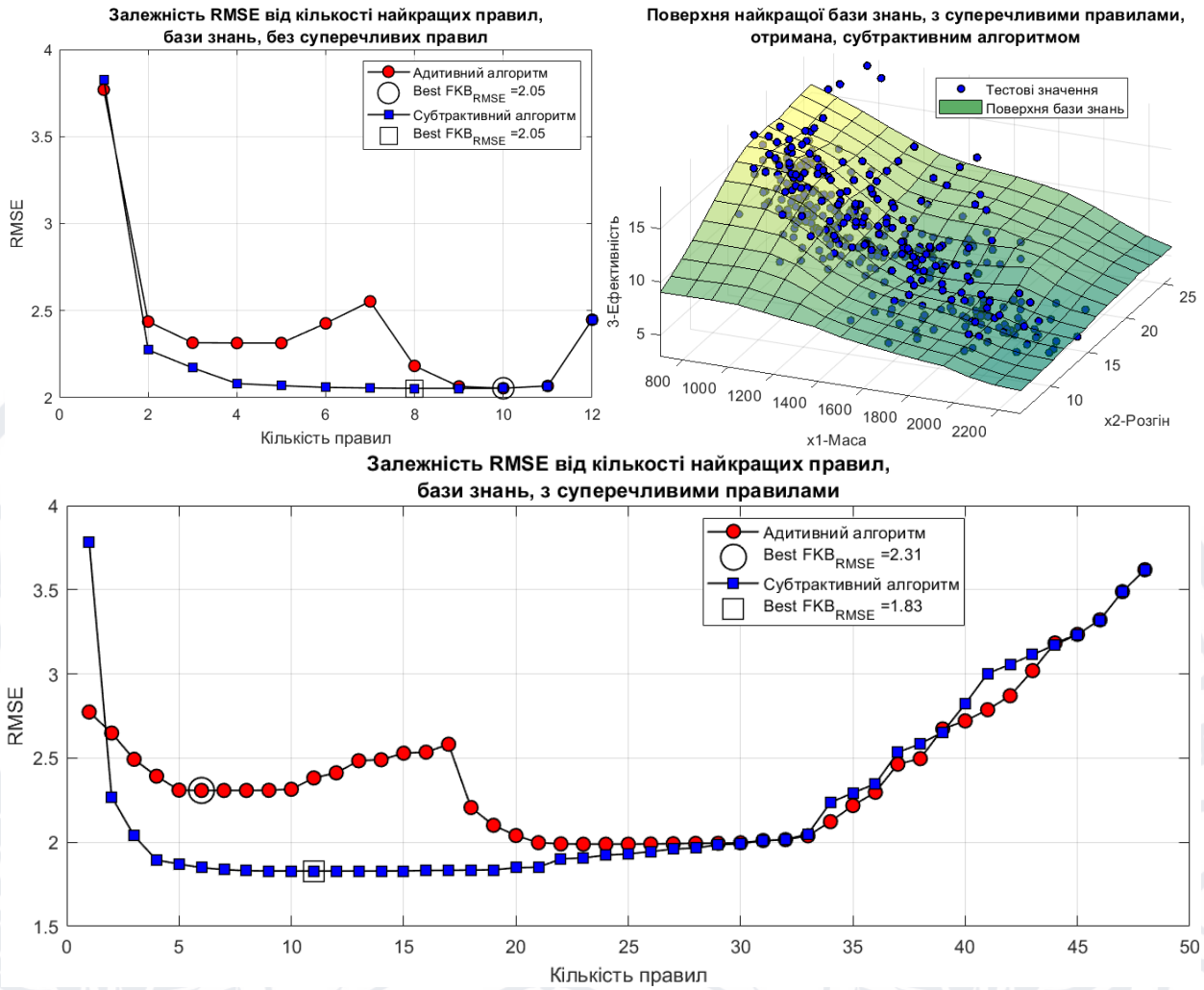


Рисунок 4.16 – Криві навчання баз знань з та без суперечливих правил залежності (г) , поверхня найкращої бази знань

Майже в усіх експериментах нечіткі бази знань з суперечливими правилами кращі за точністю та якістю відновлюваної залежності ніж звичайні нечіткі бази знань.

Для наочності на рис. 4.17, 4.18, 4.19 виведено кілька поверхонь залежності (в) (суттєво нелінійної), отриманих адитивним алгоритмом, для суперечливих та несуперечливих баз знань однакового обсягу. З цих рисунків видно, що нечіткі бази знань не лише мають менше *RMSE*, але і візуально краще відновлюють залежність.

З рис. 4.17 видно, що компактна нечітка база знань з суперечливими правилами набагато краще описує задану залежність. В компактну нечітку базу знань з суперечливими правилами не потрапило жодного суперечливого

правила табл. 4.5, але точність даної нечіткої бази знань краща за точність нечіткої бази знань без суперечливих правил. Можливо, це пояснюється тим, що при початковому відборі правил-кандидатів були обрані не найкращі правила з всієї множини. Тому використання суперечливих правил дозволяє не лише підвищувати роздільну здатність, а й частково нівелювати помилки експерта по відбору найкращих правил.

Таблиця 4.5 – Правила найкращої компактної (4 правила) нечіткої бази знань, залежності (в)

Без суперечливих правил			З суперечливими правилами		
Якщо		Тоді	Якщо		Тоді
x_1	x_2	y	x_1	x_2	y
Середній	Дуже низький	Середній	Низький	Середній	Середній
Низький	Дуже низький	Середній	Низький	Високий	Вище середнього
Високий	Дуже низький	Середній	Високий	Високий	Середній
Дуже високий	Дуже низький	Вище середнього	Дуже високий	Дуже високий	Низький

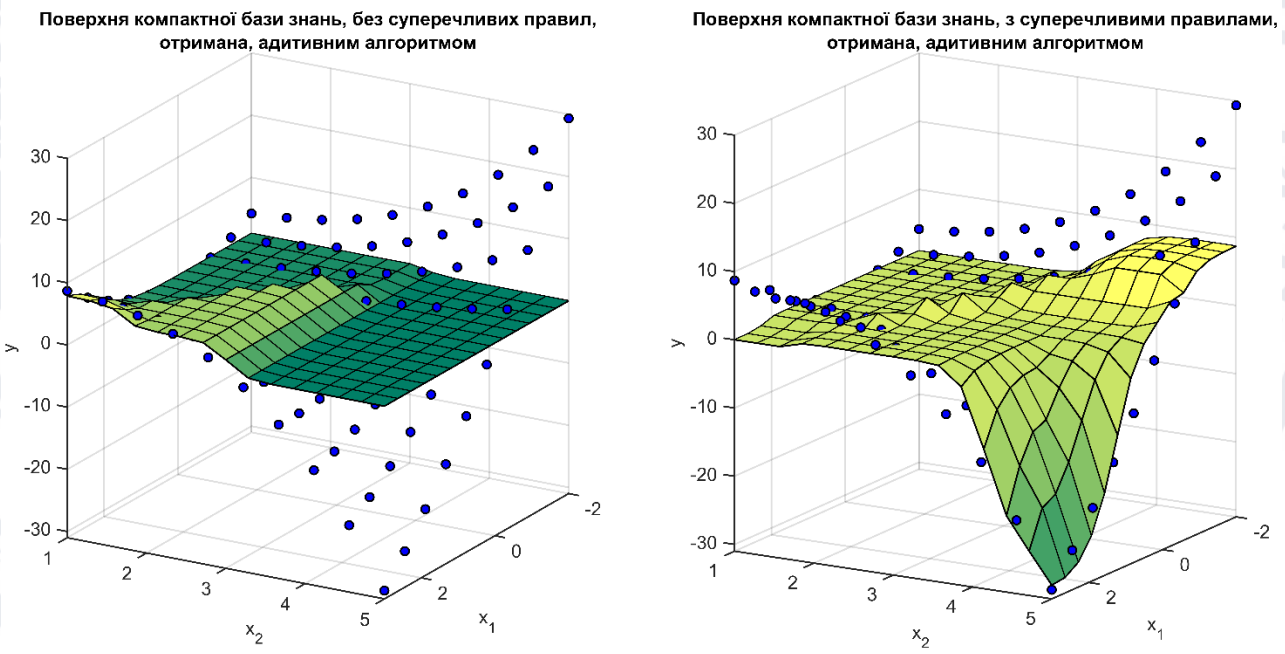


Рисунок 4.17 – Поверхні компактних баз знань (4 правила) залежності(в), з нанесеними на них тестовими значеннями

З рис. 4.18 видно, що компактна нечітка база знань з суперечливими правилами набагато знов краще описує задану залежність. В середню нечітку

базу знань з суперечливими правилами потрапило дві пари суперечливих правил табл. 4.6, це певною мірою покращило точність. Також не слід забувати що в нечітку базу знань експертно було відібрано не найкращі правила. Тому і точність між цими нечіткими базами знань значна.

Таблиця 4.6 – Правила найкращої середньої (10 правил) нечіткої бази знань, залежності (в)

Без суперечливих правил			З суперечливими правилами		
Якщо		Тоді	Якщо		Тоді
x_1	x_2	y	x_1	x_2	y
Середній	Дуже низький	Середній	Низький	Середній	Середній
Низький	Дуже низький	Середній	Низький	Високий	Вище середнього
Високий	Дуже низький	Середній	Високий	Високий	Середній
Дуже високий	Дуже низький	Вище середнього	Дуже високий	Дуже високий	Низький
Високий	Низький	Середній	Низький	Дуже високий	Високий
Низький	Низький	Вище середнього	Середній	Дуже високий	Середній
Середній	Низький	Середній	Середній	Середній	Середній
Високий	Середній	Середній	Високий	Дуже високий	Низький
Низький	Середній	Вище середнього	Середній	Дуже високий	Вище середнього
Дуже високий	Дуже високий	Низький	Низький	Середній	Вище середнього

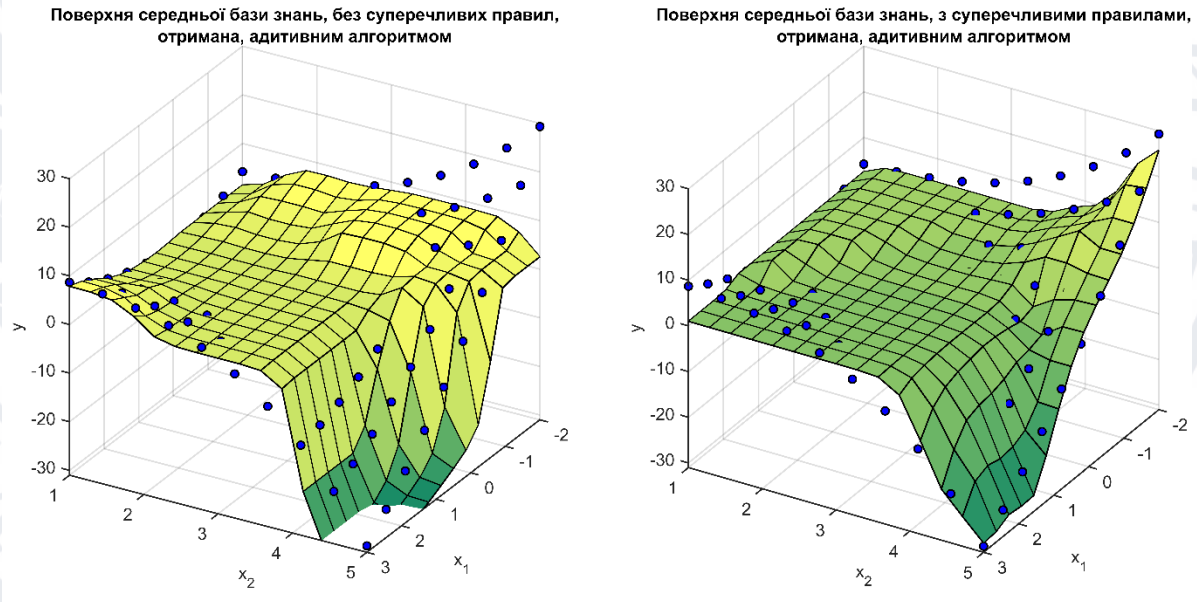


Рисунок 4.18 – Поверхні середніх баз знань (10 правил) залежності(в), з нанесеними на них тестовими значеннями

Нечітка база знань див. рис. 4.19 з та без суперечливих правил приблизно однаково описують задану залежність(в). Підтвердження

висновку про відбір суперечливих правил подано на табл. 4.7, видно, що до найкращої нечіткої бази знань потрапила значна кількість суперечливих правил, що покращило розділову здатність і покращило точність самої нечіткої бази знань.

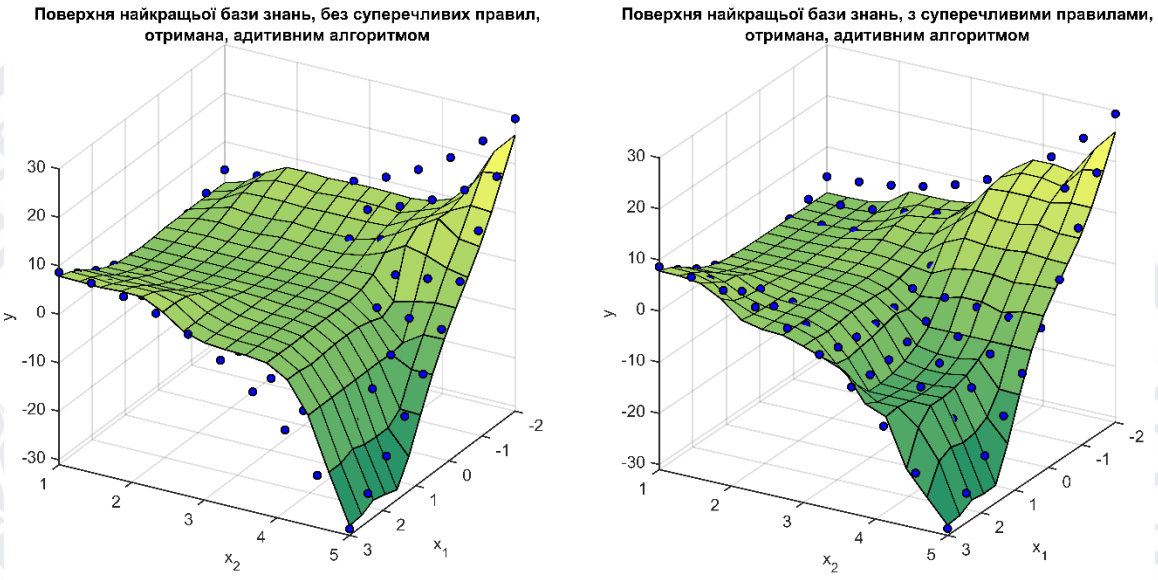


Рисунок 4.19 – Поверхні найкращих баз знань залежності (в), з нанесеними на них тестовими значеннями

Таблиця 4.7 – Правила найкращої нечіткої бази знань з суперечливими правилами, для залежності (в)

Якщо		Тоді
x_1	x_2	y
Низький	Середній	Середній
Низький	Високий	Вище середнього
Високий	Високий	Середній
Дуже високий	Дуже високий	Низький
Низький	Дуже високий	Високий
Середній	Дуже високий	Середній
Середній	Середній	Середній
Високий	Дуже високий	Низький
Середній	Дуже високий	Вище середнього
Низький	Середній	Вище середнього
Середній	Низький	Середній
Дуже високий	Дуже низький	Вище середнього
Високий	Дуже низький	Середній
Високий	Низький	Середній
Високий	Середній	Середній
Високий	Високий	Низький
Середній	Високий	Вище середнього
Дуже високий	Середній	Середній
Низький	Високий	Високий
Середній	Високий	Середній

До того ж, з табл. 4.7 видно, що в базі знань, консеквенти суперечливих правил мають сусідні функції належності, що є логічно. Тому можна покращити швидкодію жадібного адитивного алгоритму, зменшивши можливий простір вибору, скоротивши кількість кандидатів. А саме: при відборі якогось з правил, в наступних ітераціях розглядати лише сусідні в частині консеквенту, а інші відкидати.

Для прикладу див. табл. 4.8, першим в нас відібралось правило: «якщо x_1 = Низький та x_2 = Середній то y = Середній», доцільно розглядати правила з консеквентами *низький* та *вище середнього*, оскільки лише вони, можливо, покращать точність.

Таблиця 4.8 – Частина бази знань з суперечливими правилами з однаковим антецедентами та різними консеквентами

Якщо		Тоді
x_1	x_2	y
Низький	Середній	Низький
Низький	Середній	Середній
Низький	Середній	Вище середнього
Низький	Середній	Високий

Після запропонованих модифікацій було проведено комп'ютерні експерименти в яких визначалась кількість викликів основної функції (найбільш ресурсо-затратна). І як виявилось запропонований підхід має сенс, з табл. 4.9 видно, що є скорочення кількості операцій вдвічі, а в окремих випадках майже втричі.

Також встановлена закономірність між кількістю терм-множин вихідної змінної та покращенням швидкодії. Чим більше терм-множин вихідної змінної тим біль результативнішим буде покрашення в плані ресурсів.

Таблиця 4.9 - Кількість викликів основної функції

Залежність	а	б	в	г	
Алгоритм синтезу суперечливої бази знань	3240	1830	3240	1176	Кількість викликів основної функції
Метод найближчого сусіда	1180	694	1457	591	
Кількість терм-множин вихідної змінної	5	5	4	4	

4.5 Синтез несуперечливих баз знань з підмножини суперечливих правил

Іноді виникає потреба в синтезі баз знань без суперечливих правил, в тому випадку коли потрібна максимальна інтерпретація бази знань. Тому було прийнято рішення модифікувати алгоритм синтезу баз знань з суперечливими правилами шляхом накладання обмеження, а саме: при відборі якогось з правил відразу відкидати з такими ж антецедентами та відмінними консеквентами.

Криві навчання подано на рис. 4.20, 4.21, 4.22, 4.23. В усіх експериментах нечіткі бази знань синтезовані даним алгоритмом гірші за бази знань з суперечливими правилами, однак в деяких випадках кращі, або близькі до баз без суперечливих правил. Головною перевагою даного алгоритму є те, що можна отримати базу знань прийнятної точності, без експертних знань.

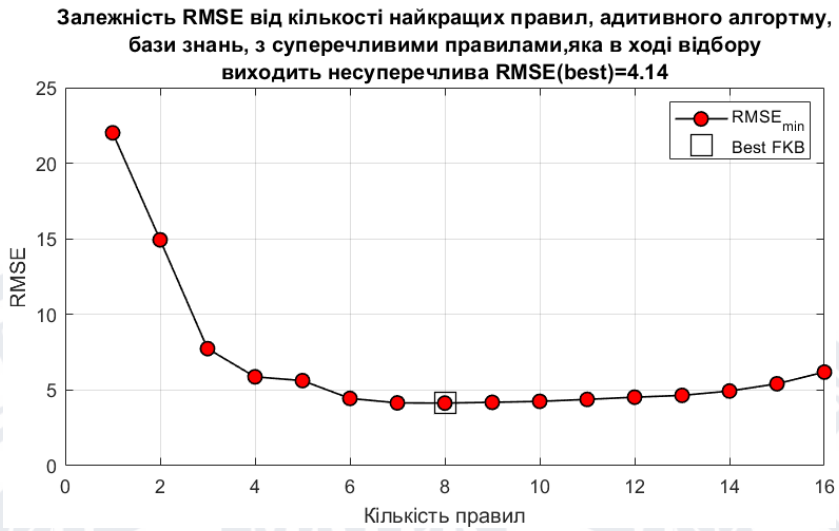


Рисунок 4.20 – Криві навчання баз знань залежності (а)

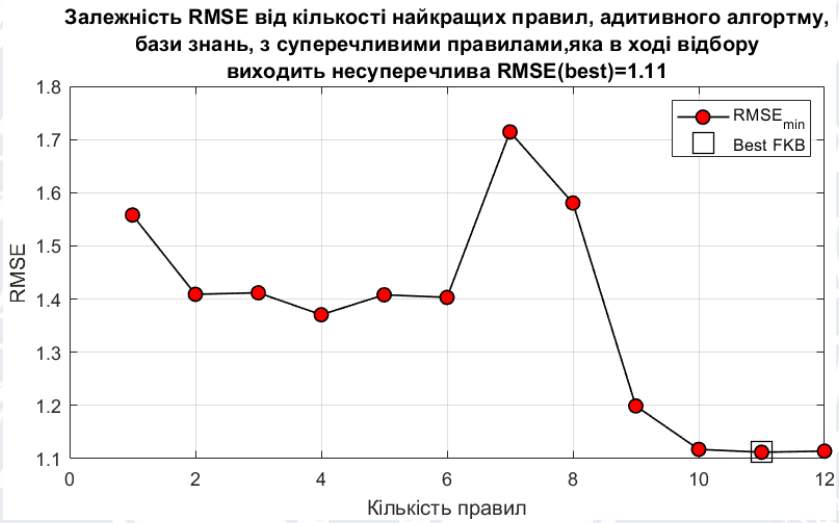


Рисунок 4.21 – Криві навчання баз знань залежності (б)

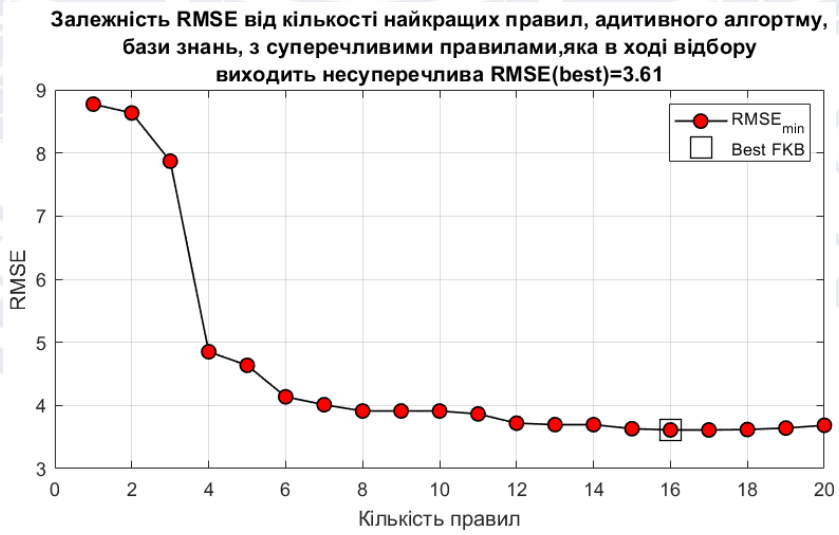


Рисунок 4.22 – Криві навчання баз знань залежності (в)

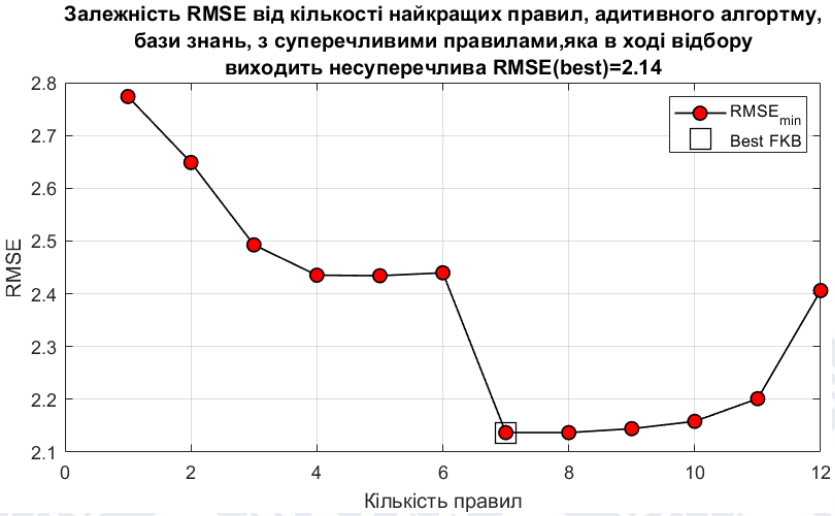


Рисунок 4.23 Криві навчання баз знань залежності (г)

4.6 Класичний алгоритм Ванга-Менделя для синтезу нечітких правил баз знань та запропоновані варіанти покращення алгоритму

Класичним методом синтезу нечітких правил баз знань є алгоритм Ванга-Менделя, він доволі простий та дієвий. Провівши деякі модифікації, його можливо покращити і отримати значний приріст в точності ідентифікації.

Синтезуємо правила для нечітких баз знань алгоритмом Ванга-Менделя та його модифікацій, проведемо відбір правил з цих баз знань та побудуємо криві залежності точності ідентифікації від кількості правил. Криві навчання подано на рис. 4.24, 4.25, 4.26, 4.27.

Для близької до лінійної залежності (а) покращення за рахунок використання модифікованих алгоритмів Ванга-Менделя, а саме головного конкурента, значне, з 4.26 до 3.5, див. рис. 4.24, але не краще в порівняні з методом синтезу нечітких баз знань з суперечливими правилами в показник точності складає 2.65.

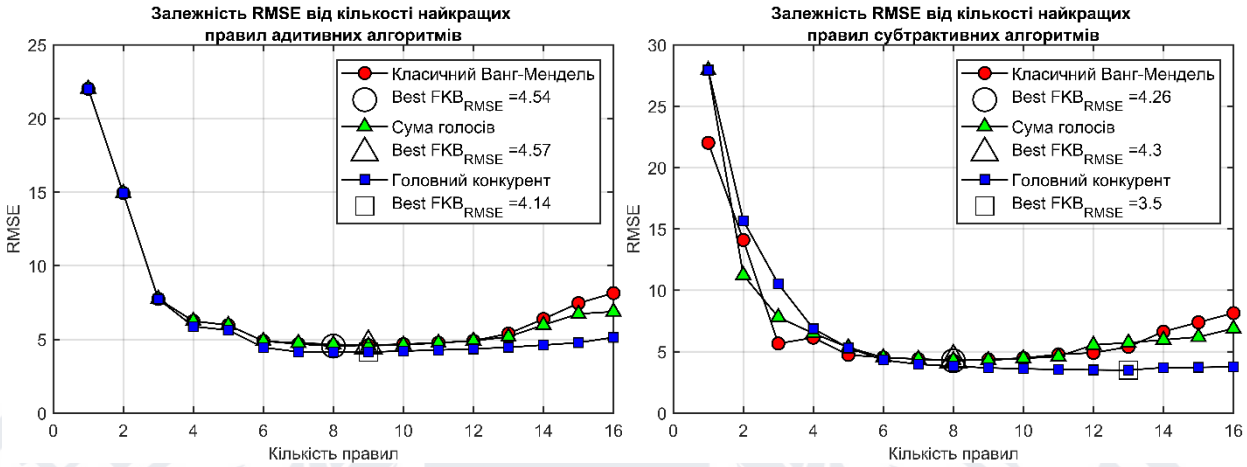


Рисунок 4.24 – Криві навчання баз знань залежності (а)

Для суттєво нелінійної залежності (б) покращення за рахунок використання модифікованих алгоритмів Ванга-Менделя, а саме суми голосів, не значне, з 1.09 до 1.02, див. рис. 4.25, також, не вдалося покращити показник точності методу синтезу нечітких баз знань з суперечливими правилами, в якого він складає 1.02.

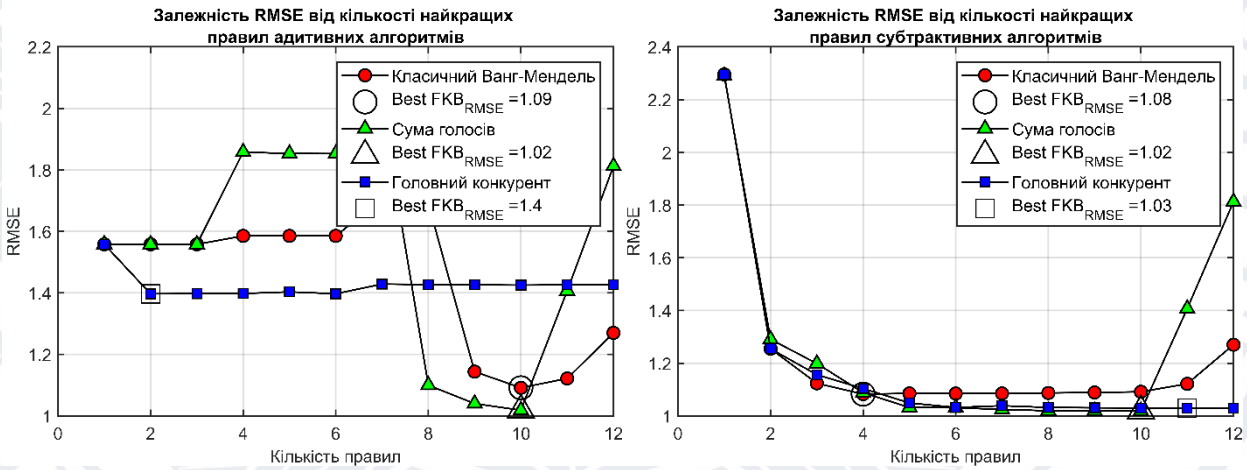


Рисунок 4.25 – Криві навчання баз знань залежності (б)

Для суттєво нелінійної залежності (в) покращення за рахунок використання модифікованих алгоритмів Ванга-Менделя, а саме суми голосів, практично не відбулось, з 3.35 до 3.29, див. рис. 4.26, до того ж він значно гірший за показник точності методу синтезу нечітких баз знань з суперечливими правилами, в якого він складає 2.41.

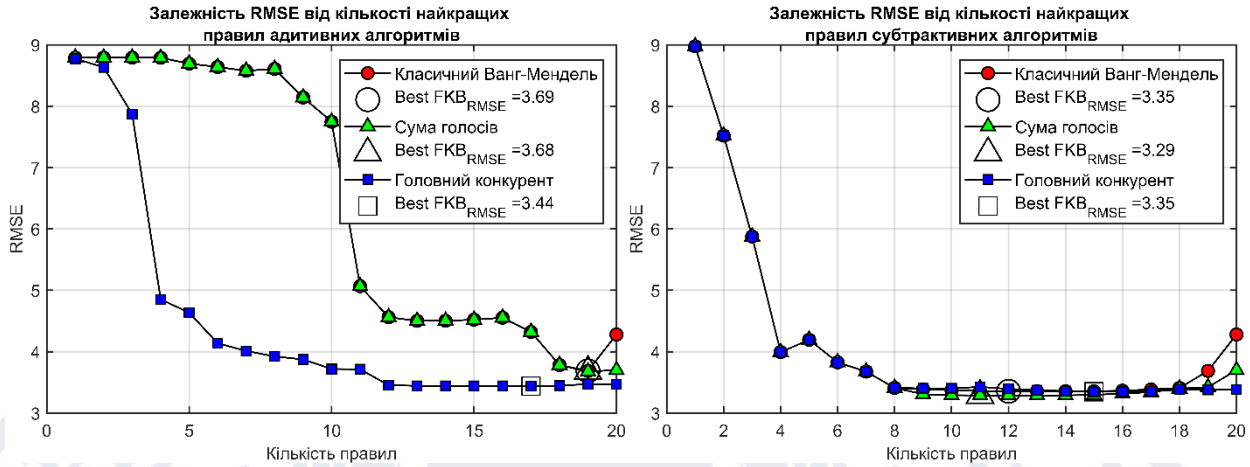


Рисунок 4.26 – Криві навчання баз знань залежності (в)

Для близької до лінійної залежності (г) покращення за рахунок використання модифікованих алгоритмів Ванга-Менделя, не відбулось див. рис. 4.27.

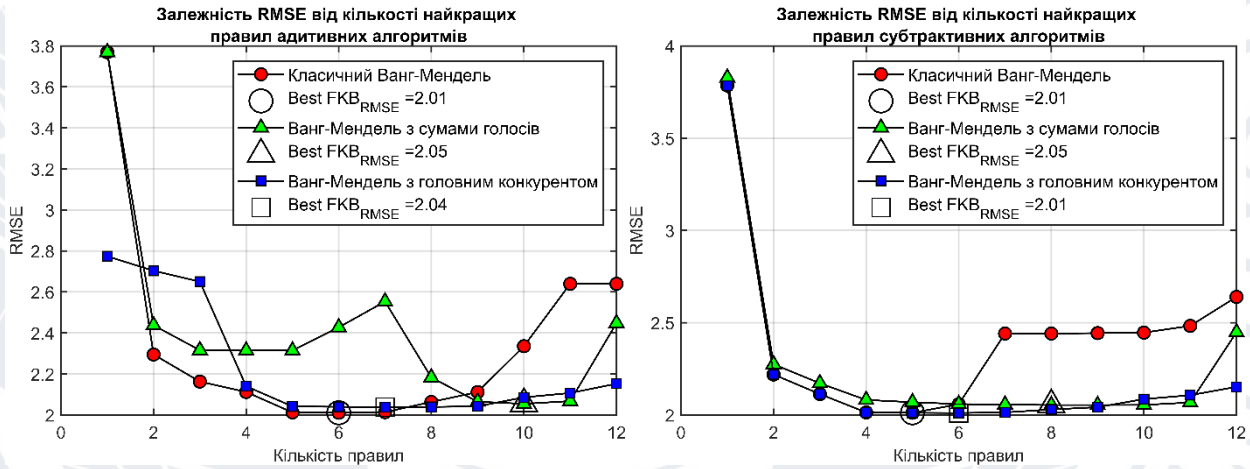


Рисунок 4.27 – Криві навчання баз знань залежності (г)

Запропоновані модифікації майже в усіх залежностях кращі за класичний алгоритм Ванга-Менделя, а в деяких випадках краще за метод синтезу нечітких баз знань з суперечливими правилами.

4.7 Якість досліджених методів

Для узагальнення результати експериментів, а саме точності ідентифікації, було систематизовано та зведено в табл. 4.10. Як видно в 3 з 4 експериментів бази знань з суперечливими правилами мають кращий показник точності ніж бази знань без суперечливих правил. Також видно, що

модифікація алгоритму Ванга-Менделя з головним конкурентом (суперечливі правила) також показує непогані результати в порівняні з подібними методами, десь точність значно краща, десь така ж.

Таблиця 4.10 – Показник точності алгоритмів залежностей (а-г)

		Залежність (а)	Залежність (б)	Залежність (в)	Залежність (г)
		RMSE			
Синтез несуперечливої бази знань	Адитивний	4,57	1,02	3,68	2,05
	Субтрактивний	4,3	1,02	3,29	2,05
Синтез суперечливої бази знань, найближчого сусіда	Адитивний	4,14	1,29	2,41	2,31
	Субтрактивний	2,65	1,03	4,04	1,83
Синтез несуперечливої бази знань з суперечливої	Адитивний	4,14	1,11	3,61	2,14
Класичний Ванг-Мендель	Адитивний	4,54	1,09	3,69	2,01
	Субтрактивний	4,26	1,08	3,35	2,01
Модифікований Ванг-Мендель з сумами голосів	Адитивний	4,57	1,02	3,68	2,05
	Субтрактивний	4,3	1,02	3,29	2,05
Модифікований Ванг-Мендель з головним конкурентом	Адитивний	4,14	1,4	3,44	2,04
	Субтрактивний	3,5	1,03	3,35	2,01

Висновки до розділу 4

В даному було проведено обчислювальні експерименти із синтезу нечітких баз знань Мамдані з та без суперечливих правил, встановлено, що застосування таких суперечливих правил підвищує «роздільну здатність» опису залежності, яку моделює база знань. Відповідно, точність нечіткої бази знань за рахунок використання суперечливих правил зростає.

ВИСНОВКИ

В ході виконання даної роботи були проаналізовані наявні методи синтезу та відбору правил нечітких баз знань. Було визначено їхні переваги та недоліки, і на основі цього розроблені власні алгоритми відбору правил нечіткої бази знань, покращено класичний алгоритм синтезу правил нечітких баз знань Ванга-Менделя шляхом використання суперечливих правил, а саме головного конкурента. Також розроблені власні алгоритми із застосуванням суперечливих правил. Експериментально встановлено, що застосування даних алгоритмів та підходів шляхом використання суперечливих правил, за рахунок підвищення «роздільної здатності» опису, забезпечує суттєву кращу точність. Це дозволяє досягнути бажаної точності нечіткої бази знань за меншої кількості правил.

Подальші роботи спрямовуватимуться на дослідження питань навчання нечітких баз знань з суперечливими правилами, тобто питанням параметричної ідентифікації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ТА ЛІТЕРАТУРИ

1. Резнік Р.Ю., Штовба С.Д. Синтез нечіткої бази знань Мамдані з використанням суперечливих правил. Матеріали III Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих вчених "Прикладні інформаційні технології" (22 квітня 2022 року) - Вінниця: ДонНУ імені Василя Стуса., с. 34-37.

2. Резнік Р.Ю., Штовба С.Д. Модифікація алгоритму Ванга-Менделя Шляхом використання суперечливих правил. Матеріали III Всеукраїнської науково-практичні конференції «Комп'ютерні технології обробки даних» (8 грудня 2022 року) - Вінниця: ДонНУ імені Василя Стуса (подано до публікації).

3. Резнік Р.Ю., Штовба С.Д. Синтез нечітких баз знань Мамдані з використанням суперечливих правил. «Вісник Донецького національного університету імені Василя Стуса, «Комп'ютерні науки та кібер-фізичні системи» (подано до публікації).

4. Dragan Perakovic, Ivan Grgurevic, Vladimir Remenar Possibility of applying fuzzy logic in the e-Learning system. Conference proceedings of CECiiS 2008, Central European Conference on Information and Intelligent Systems

5. Daniel Doz, Darjo Felda, Mara Cotic Using Fuzzy Logic to Assess Students' Mathematical Knowledge. Conference: Nauka i obrazovanje – izazovi i perspektive 2022. Ст.263–278.

6. Штовба С.Д., Галущак А.В. Ідентифікація багатофакторних залежностей за допомогою баз знань. Лабораторний практикум: електронний навчальний посібник. – Вінниця: Вінницький національний технічний університет, 2016. – 96 с.

7. Serhiy Shtovba, Viktor Mazurenko, Mykola Petrychko Information Technology for Extracting the Accurate, Compact and Interpretable Mamdani-type Rule Base. CEUR Workshop Proceedings "Information Control Systems & Technologies". 2020. Ст.386-400.

8. Fuzzy Logic: An Introductory Course for Engineering Students, Enric Trillas, Luka Eciolaza, Springer, 2015, 215 с.

9. Introduction to Fuzzy Logic 1st Edition, James K. Peckol, Wiely, 2021, 304 с.

10. Математичний апарат штучного інтелекту в електроенергетичних системах: підручник / В. В. Кирик.– Київ : КПІ ім. Ігоря Сікорського, Вид-во «Політехніка» 2019.– 224с.

11. С.Д. Штовба В.В. Мазуренко Інтелектуальні технології ідентифікації залежностей. Лабораторний практикум Вінницький національний технічний університет. 2014. Ст.48.

12. СД Штовба Моделювання залежностей за допомогою нечіткої бази знань з нечіткими регресійними рівняннями. Вісник Вінницького політехнічного інституту 3 випуск. Ст. 195-199.

13. С.Д. Штовба, А.В. Галущак Быстрое обучение нечеткого классификатора по методу главных конкурентов. Международный научно-практический форум «Наука и бизнес». 2016. Ст.153–161.

14. СД Штовба, ВВ Мазуренко, РО Тылец Информационная технология нечеткой идентификации для синтеза точных, компактных и интерпретабельных баз знаний. Computer Science and Telecommunications. 2016. Ст. 8-22.

15. Fuzzy Sets, Fuzzy Logic and Their Applications, Michael Gr Voskoglou, Mdpi AG, 2020, 366c.

16. М.М. Демчина Використання нечітких правил для подання знань в інтелектуальних системах нафтогазової предметної області. Контроль, автоматика та електротехніка 2012. ст. 132-134.

17. Fuzzy Logic Models and Fuzzy Control: An Introduction, D.S. Hooda, Vivek Raich, Alpha Science International, 2017, 408c.

18. Ротштейн А.П. Интеллектуальные технологии идентификации / А.П. Ротштейн. – Винница: Вінниця–УНІВЕРСУМ, 1999. – 320 с.

19. Applications of Fuzzy Logic in Planning and Operation of Smart Grids, Mehdi Rahmani-Andebili, Springer, 2021, 229c.

20. Капустинський Роман Ігорович. Алгоритми нечіткого управління системою доступу до бази даних підприємства / Algorithms for fuzzy control of the enterprise database access system. Архів кваліфікаційних робіт Західноукраїнського національного університету 2020 – Тернопіль. Ст.26,51,52,65.

21. Fuzzy Logic: Recent Applications and Developments, Jenny Carter, Francisco Chiclana, Arjab Singh Khuman, Tianhua Chen, Springer, 2021, 278c.

22. Ishibuchi H. Selecting fuzzy if-then rules for classification problems using genetic algorithms / H. Ishibuchi, K. Nozaki, N. Yamamoto, H. Tanaka // IEEE Transactions on Fuzzy Systems. – 1995. – Vol. 3, No. 3. – P. 260–270.

23. Ishibuchi H. Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems / H. Ishibuchi, T. Murata, I. B. Turksen // Fuzzy Sets and Systems. – 1997. – Vol. 89, No. 2 – P. 135–150.

24. Ishibuchi H. Three-objective genetics-based machine learning for linguistic rule extraction / H. Ishibuchi, T. Nakashima, T. Murata // Inform. Sci. – 2001. – Vol. 136, No. 1. – P. 109–133.

25. С.Д. Штовба, В.В. Мазуренко Багатокритеріальне оцінювання якості нечітких баз знань в задачах ідентифікації. Обчислювальний інтелект. – 2013. Ст.66-70.

26. Cordon O. A historical review of evolutionary learning methods for Mamdani-type fuzzy rule-based systems: Designing interpretable genetic fuzzy systems / Cordon O. // International Journal of Approximate Reasoning. – 2011. – Vol. 52 – P. 894–913.

27. Cordon O. Ten years of genetic fuzzy systems: current framework and new trends / O. Cordon, F. Gomideb, F. Herreraa, F. Homannc, L. Magdalenad // Fuzzy Sets and Systems. – 2004. – Vol. 141 – P. 5–31.

28. Штовба С.Д. Генетичний алгоритм вибору правил нечіткої бази знань, збалансованої за критеріями точності та компактності / Штовба С.Д., Мазуренко В.В., Савчук Д.А. // Наукові праці Вінницького національного технічного університету. – 2012. – №3. – Режим доступу: http://praci.vntu.edu.ua/article/view/2335/2603.

29. Shtovba S., Mazurenko M., Petrychko M. Information Technology for Extracting the Accurate, Compact and Interpretable Mamdani-type Rule Base // CEUR Workshop Proceedings, Vol. 2711 “Proc. of the XI Information Control Systems & Technologies”. – 2019. – P. 386-400.

30. Штовба С.Д. Вплив кількості нечітких правил на точність бази знань Мамдані/ Штовба С.Д., Мазуренко В.В., Панкевич О.Д. // Вісник Хмельницького національного університету. Технічні науки. – 2011. – №2. – С. 185–188.

31. Wang L. X., Mendel J. M. Generating fuzzy rules by learning from examples //IEEE Transactions on systems, man, and cybernetics. – 1992. – Т. 22. – №. 6. – С. 1414-1427.

32. Minshen Hao, Jerry M Mendel Extracting IF-THEN Rules from Numerical Data using Wang-Mendel Methods 2013.

33. Jin Gou, Zongwen Fan, Cheng Wang, Haixiao Chi, Wei Luo, An improved Wang-Mendel method based on the FSFDP clustering algorithm and sample correlation //Journal of Intelligent and Fuzzy Systems. – 2016. – С. 2839-2850

34. Javier Andreu-Perez, Prashant Gupta Enhanced Type-2 Wang-Mendel Approach // Journal of Experimental & Theoretical Artificial Intelligence. – 2022.

35. Jin Gou, Feng Hou, Wenyu Chen, Cheng Wang, Improving Wang–Mendel method performance in fuzzy rules generation using the fuzzy C-means clustering algorithm //Neurocomputing. – 2015. – С.1293-1304.

36. J. Gou, W.-Y. Chen An improved Wang-Mendel method based on cooperation degree of sample and self-organizing mapping //Moshi Shibie yu Rengong Zhineng/Pattern Recognition and Artificial Intelligence. – 2013. – С.1079-1088.

37. Francisco Herrera, Francisco Herrera A proposal for improving the accuracy of linguistic modeling // IEEE Transactions on Fuzzy Systems – 2000. – 8(3). – С. 335 – 344.

38. Diego Álvarez-Estévez, V. Moret-Bonillo Revisiting the Wang–Mendel algorithm for fuzzy classification // Expert Systems – 2018. – С. 35.

39. W. Chen, J. Gou Improved wang-mendel scheme based on cooperation among input variables // International Review on Computers and Software. – 2012. – С.2685-2689.

40. Применение метода нечеткой логики с помощью пакета MATLAB – Режим доступу до ресурсу: https://studbooks.net/2180357/informatika/primenenie_metoda_nechetkoy_logiki_pomoschyu_paketa_matlab (дата звернення 14.08.2022).

41. Cryptography and Cryptanalysis in MATLAB: Creating and Programming Advanced Algorithms, Marius Iulian Mihailescu, Stefania Loredana Nita, Apress, 2021, 208 p..

42. Practical MATLAB Deep Learning: A Project-Based Approach, Michael Paluszek, Stephanie Thomas, Apress, 2020, 348 p.

43. Ротштейн А. П. Проектування нечітких баз знань: лабораторний практикум та курсове проектування : [навчальний посібник] / А. П. Ротштейн, С. Д. Штовба - Вінниця : ВНТУ, 1999. – 65 с.

44. Гоблик Н. М., Гоблик В. В. MATLAB в інженерних розрахунках. Комп’ютерний практикум – Львів: Львівська політехніка, 2020. – 192с.

45. MATLAB for Beginners: A Gentle Approach, Peter Kattan, Petra Books, 2021, 334 c.

46. С. Д. Штовба Методи оптимізації в середовищі MatLab. Лабораторний практикум. Навчальний посібник. - Вінниця: ВДТУ, 2001.-56с.

47. Matlab and Python Programming: A Practical Guide for Engineers & Data Scientists, Upskill Learning, CreateSpace Independent Publishing Platform, 2016, 272c.

48. MATLAB for Machine Learning: Practical examples of regression, clustering and neural networks, Giuseppe Ciaburro, Università degli Studi della Campania "Luigi Vanvitelli", 2017, 382c.

49. Learning MATLAB: A Problem Solving Approach, Walter Gander, Springer, 2015, 163 c.

50. Auto MPG Data Set – Режим доступу до ресурсу: https://archive.ics.uci.edu/ml/datasets/auto+mpg (дата звернення 14.08.2022).

ДОДАТКИ

ДОДАТОК А

Лістинг програми

```
clear
warning('off')
set(0, 'DefaultLineLineWidth', 1);
set(0,'defaultAxesFontSize',11)
tic
%Побудова графіка функції y = (x1^2 + x2) / (0.1*exp^x1 + x2)
%в області x1є[-5,5] та x2є[3,7].
left_lim_x1=-5;right_lim_x1=5;
left_lim_x2=3;right_lim_x2=7;
f = @(x1, x2) ((x1.^2)+x2)./(0.1*exp(x1)+x2); %функція
n=15;  %кількість точок дискретизації.
%Розбиваємо інтервал на n рівних відрізків:
range1 = linspace(left_lim_x1, right_lim_x1, n);
range2 = linspace(left_lim_x2, right_lim_x2, n);

%Створення декартової сітки nxn, по якій побудуємо графік:
[x1, x2] = meshgrid(range1, range2);
y=f(x1,x2);

lim_z = [round(min(min(y)))-1 round(max(max(y)))+1];

fis_selected = readfis('exp_2_selected_rules_2.fis'); % зчитуємо базу знань
Nrules = length(fis_selected.rule); % Визначаю кількість правил
line = 1:Nrules;
fis_all = readfis('exp_2_all_rules_2.fis');
Nrules_long = length(fis_all.rule); % Визначаю кількість правил суперечливої бази
знань
line_long = 1:Nrules_long;

num_output_mf = length(fis_all.Outputs.MembershipFunctions);

% Формування тестових значень та нанесення їх на поверхню еталонної залежності
range_x1 = linspace(left_lim_x1, right_lim_x1,10);
range_x2 = linspace(left_lim_x2, right_lim_x2,10);
x1_points = zeros([1,100]);
x2_points = x1_points;
output = x1_points;
for i=1:10
    for j=1:10
        output((i-1)*10+j)=f(range_x1(i),range_x2(j));
        x1_points(i*10-10+j) = range_x1(i);
        x2_points(i*10-10+j) = range_x2(j);
    end
end
[x1, x2] = meshgrid(range1, range2);
y=f(x1,x2);
input = cat(1,x1_points,x2_points);
data = cat(1,x1_points,x2_points,output)';

% Поверхня еталонної залежності, з нанесеною на неї тестовими значеннями
figure;
surf(x1,x2,y,'edgecolor','none');
xlabel('x_1');
ylabel('x_2');
zlabel('y')
xlim(minmax(range1));
ylim(minmax(range2));
zlim(minmax(lim_z));
colormap('summer(20)')
hold on
plot3(x1_points,x2_points,output,'go','LineStyle','none','MarkerSize',5,...
    'LineWidth',1,'MarkerEdgeColor','k','MarkerFaceColor','b');
```

```
title(['Поверхня еталонної залежності,',newline,'з нанесеною на неї тестовими
значеннями'])
legend("Поверхня еталонної залежності","Тестові значення");
    alpha(.8)


% Адитивний алгоритм, бази знань, правила в якій не суперечать одне одному
% Субтрактивний алгоритм, бази знань, правила в якій не суперечать одне одному
[MIN_RMSE_sel_add,MIN_IND_sel_add,Min_RMSE_Value_sel_add,Min_Ind_sel_add] = ...
func_add(fis_selected,line,line,Nrules,input,output);

[MIN_RMSE_sel_sub,MIN_IND_sel_sub,Min_RMSE_Value_sel_sub,Min_Ind_sel_sub] = ...
func_sub(fis_selected,line,line,Nrules,input,output);

graph_1_line(line,MIN_RMSE_sel_add,Min_Ind_sel_add,Min_RMSE_Value_sel_add,...
    "адитивного алгортму","без суперечливих правил")
graph_1_line(line,MIN_RMSE_sel_sub,Min_Ind_sel_sub,Min_RMSE_Value_sel_sub,...
    "субтрактивного алгортму","без суперечливих правил")

% Поверхні найкращих баз знань без суперечливих правил
graph_surf(line,lim_z,Min_Ind_sel_add,MIN_IND_sel_add,fis_selected,...
    "адитивним алгоритмом","без суперечливих правил",x1_points,x2_points,output)
graph_surf(line,lim_z,Min_Ind_sel_sub,MIN_IND_sel_sub,fis_selected,...
    "субтрактивним алгоритмом","без суперечливих правил",x1_points,x2_points,output)


% Порівняння показника точності адитивного та субтрактивного алгоритмів
% бази знань з не суперечливими правилами
graph_2_line(line,MIN_RMSE_sel_add,Min_Ind_sel_add,Min_RMSE_Value_sel_add,"Адитивний
алгоритм",...
    "без суперечливих
правил",MIN_RMSE_sel_sub,Min_Ind_sel_sub,Min_RMSE_Value_sel_sub,"Субтрактивний
алгоритм")

% Адитивний алгоритм, бази знань, правила в якій суперечать одне одному
% Субтрактивний алгоритм, бази знань, правила в якій суперечать одне одному
[MIN_RMSE_all_add_conf,MIN_IND_all_add_conf,Min_RMSE_Value_all_add_conf,...
Min_Ind_all_add_conf] = func_add(fis_all,line,line_long,Nrules,input,output);

[MIN_RMSE_all_sub_conf,MIN_IND_all_sub_conf,Min_RMSE_Value_all_sub_conf,...
Min_Ind_all_sub_conf] = func_sub(fis_all,line,line_long,Nrules_long,input,output);

% Порівняння показника точності адитивного та субтрактивного
% алгоритмів, бази знань з суперечливими правилами
graph_2_line(line_long,MIN_RMSE_all_add_conf,Min_Ind_all_add_conf,Min_RMSE_Value_all_a
dd_conf,...
    "Адитивний алгоритм","з суперечливими
правилами",MIN_RMSE_all_sub_conf,Min_Ind_all_sub_conf,...
    Min_RMSE_Value_all_sub_conf,"Субтрактивний алгоритм")

% Поверхні найкращих баз знань без суперечливих правил
graph_surf(line_long,lim_z,Min_Ind_all_add_conf,MIN_IND_all_add_conf,fis_all,...
    "адитивним алгоритмом","з суперечливими правилами",x1_points,x2_points,output)
graph_surf(line_long,lim_z,Min_Ind_all_sub_conf,MIN_IND_all_sub_conf,fis_all,...
    "субтрактивним алгоритмом","з суперечливими правилами",x1_points,x2_points,output)

% Адитивний алгоритм з найближчим сусідом, бази знань, правила в якій суперечать одне
одному
[MIN_RMSE_all_add_conf_neighbor,MIN_IND_all_add_conf_neighbor,Min_RMSE_Value_all_add_c
onf_neighbor,...
Min_RMSE_Ind_all_add_conf_neighbor] =
func_add_neighbor(fis_all,line,line_long,input,output);

% Залежність RMSE від кількості найкращих правил,
% алгоритму найближчого сусіда, бази знань з суперечливими правилами
figure;
plot(line,MIN_RMSE_all_add_conf_neighbor,'k o',"LineStyle",'-','MarkerSize',8,...
    'MarkerEdgeColor','k','MarkerFaceColor',[1.0,0.0,0.0]);
hold on
```

```
plot(Min_RMSE_Ind_all_add_conf_neighbor,Min_RMSE_Value_all_add_conf_neighbor,'k
s','MarkerSize',15);
legend('RMSE_m_i_n','Best FKB');
xlabel('Кількість правил')
ylabel('RMSE')
grid on
title("Залежність RMSE від кількості найкращих правил, алгоритму найближчого сусіда,"
+...
    newline + "бази знань з супечливими правилами RMSE(best)=" +
num2str(Min_RMSE_Value_all_add_conf_neighbor,3));


% Адитивний алгоритм, бази знань, правила в якій суперечать одне одному,
% однак, в ході відбору вона виходить несуперечлива
[MIN_RMSE_all_add_NoConf,MIN_IND_all_add_NoConf,Min_RMSE_Value_all_add_NoConf,...
Min_RMSE_Ind_all_add_NoConf] =
func_add_No_conf(fis_all,line,line_long,Nrules,input,output);

graph_1_line(line,MIN_RMSE_all_add_NoConf,Min_RMSE_Ind_all_add_NoConf,Min_RMSE_Value_a
ll_add_NoConf,...
    "адитивного алгортму","з суперечливими правилами,яка в ході
відбору"+newline+"виходить несуперечлива")


%Порівняння показника точності алгоритмів, які працюють з базами знань,
% з суперечливими правилами
graph_3_line(line,MIN_RMSE_all_add_conf(line),Min_Ind_all_add_conf,Min_RMSE_Value_all_
add_conf,"Адитивний алгоритм",...

MIN_RMSE_all_sub_conf(line),Min_Ind_all_sub_conf,Min_RMSE_Value_all_sub_conf,"Субтракт
ивний алгоритм",...

MIN_RMSE_all_add_NoConf,Min_RMSE_Ind_all_add_NoConf,Min_RMSE_Value_all_add_NoConf,...
    "Результуюча без суперечливих","які працюють з базами знань, з суперечливими
правилами")


% Класичний та різні модифікації Ванга-Менделя
fis_WM = WM(data,fis_selected);
% Адитивний алгоритм, Ванга-Менделя
[MIN_RMSE_WM_add,MIN_IND_WM_add,Min_RMSE_Value_WM_add,Min_Ind_WM_add] = ...
func_add(fis_WM,line,line,Nrules,input,output);
% Субтрактивний алгоритм, Ванга-Менделя
[MIN_RMSE_WM_sub,MIN_IND_WM_sub,Min_RMSE_Value_WM_sub,Min_Ind_WM_sub] = ...
func_sub(fis_WM,line,line,Nrules,input,output);


fis_WM_sum = WM_sum(data,fis_selected);
% Адитивний алгоритм, Ванга-Менделя з сумами голосів
[MIN_RMSE_WM_sum_add,MIN_IND_WM_sum_add,Min_RMSE_Value_WM_sum_add,Min_Ind_WM_sum_add]
= ...
func_add(fis_WM_sum,line,line,Nrules,input,output);
% Субтрактивний алгоритм, Ванга-Менделя з сумами голосів
[MIN_RMSE_WM_sum_sub,MIN_IND_WM_sum_sub,Min_RMSE_Value_WM_sum_sub,Min_Ind_WM_sum_sub]
= ...
func_sub(fis_WM_sum,line,line,Nrules,input,output);


fis_WM_competitor = WM_competitor(data,fis_selected);
Nrules_competitor = length(fis_WM_competitor.rule);
line_competitor = 1:Nrules_competitor;
% Адитивний алгоритм Ванга-Менделя з головним конкурентом
[MIN_RMSE_WM_competitor_add,MIN_IND_WM_competitor_add,Min_RMSE_Value_WM_competitor_add
,...
Min_Ind_WM_competitor_add] =
func_add(fis_WM_competitor,line,line_competitor,Nrules,input,output);
% Субтрактивний алгоритм Ванга-Менделя з головним конкурентом
[MIN_RMSE_WM_competitor_sub,MIN_IND_WM_competitor_sub,Min_RMSE_Value_WM_competitor_sub
,...
```

```
Min_Ind_WM_competitor_sub] =
func_sub(fis_WM_competitor,line,line_competitor,Nrules_competitor,input,output);

% Порівняння показника точності модифікацій алгоритмів  Ванга-Менделя
graph_3_line(line,MIN_RMSE_WM_add,Min_Ind_WM_add,Min_RMSE_Value_WM_add,"Класичний
Ванг-Мендель",...
    MIN_RMSE_WM_sum_add,Min_Ind_WM_sum_add,Min_RMSE_Value_WM_sum_add,"Сума
голосів",...

MIN_RMSE_WM_competitor_add,Min_Ind_WM_competitor_add,Min_RMSE_Value_WM_competitor_add,
...
    "Головний конкурент","адитивних алгоритмів")
graph_3_line(line,MIN_RMSE_WM_sub,Min_Ind_WM_sub,Min_RMSE_Value_WM_sub,"Класичний
Ванг-Мендель",...
    MIN_RMSE_WM_sum_sub,Min_Ind_WM_sum_sub,Min_RMSE_Value_WM_sum_sub,"Сума
голосів",...

MIN_RMSE_WM_competitor_sub,Min_Ind_WM_competitor_sub,Min_RMSE_Value_WM_competitor_sub,
...
    "Головний конкурент","субтрактивних алгоритмів")
 a_time = toc;
```

ДОДАТОК Б

Функція адитивного алгоритму

```
function [Min_RMSE,Min_Ind,Min_RMSE_Value,Min_RMSE_Ind] =
func_add(FIS,Line,Line_long,Nrules,Inp,Out)
Min_RMSE = zeros([1,Nrules]);
Min_Ind = [];
counter = 0;
%Зануляю ваги всіх правил, тобто вимключаю їх з бази знань
for j=Line_long
    FIS.rule(j).weight=0;
end
%Викликаю цикл який буде тривати до кількості правил
for i=Line_long
    RMSE_min = 10000; % Зміна яка буде зберігати мінімальне RMSE

    %Формую індекси для перебору правил, з виключенням тих правил,
    %що вже потрапили до нечіткої бази знань
    for j=setdiff(Line_long, Min_Ind)
        counter = counter + 1;
        FIS.rule(j).weight=1; %По черзі активую ваги правил
        RMSE = RMSEfis(FIS,Inp,Out); %Розраховую чергове RMSE
        %Знаходжу мінімальне RMSE та його індекс
        if RMSE_min > RMSE
            RMSE_min = RMSE;
            min_ind = j; % Зміна яка буде зберігати індекс мінімального RMSE
        end
        FIS.rule(j).weight=0; %Зануляю вагу чергового правила
    end

    FIS.rule(min_ind).weight=1;

    Min_RMSE(i) = RMSE_min; %Формую масив з RMSE, які обчислив на кожні ітерації
    Min_Ind = [Min_Ind min_ind]; %Формую черговість індексів найкращих правил
end
[Min_RMSE_Value,Min_RMSE_Ind] = min(Min_RMSE(Line)); %Точність та індекс найкращьої
нечіткої бази знань
counter
end
```

ДОДАТОК В

Функція субтрактивного алгоритму

```
function [Min_RMSE,Min_Ind,Min_RMSE_Value,Min_RMSE_Ind] =
func_sub(FIS,Line,Line_long,Nrules,Inp,Out)
for j=Line_long
    FIS.rule(j).weight=1;
end
Min_RMSE = zeros([1,Nrules]);
Min_RMSE(1) = RMSEfis(FIS,Inp,Out);
Min_Ind = [];
counter = 0;
%Викликаю цикл який буде тривати до кількості правил
for i=1:(Nrules-1)
    RMSE_min = 10000;
    %Формую індекси для перебору правил, з виключенням тих правил,
    %що вже виключили з нечіткої бази знань
    for j=setdiff(Line_long, Min_Ind)
        counter = counter + 1;
        FIS.rule(j).weight=0; %По черзі активую ваги правил
        RMSE = RMSEfis(FIS,Inp,Out); %Розраховую чергове RMSE
        %Знаходжу максимальне RMSE та його індекс
        if RMSE < RMSE_min
            RMSE_min = RMSE;
            max_ind = j; % Зміна яка буде зберігати індекс максимального RMSE
        end
        FIS.rule(j).weight=1; %Вимикаю відпрацьоване правило
    end

    FIS.rule(max_ind).weight=0;
    Min_Ind = [Min_Ind max_ind]; %Формую черговість індексів найгірших правил правил
    Min_RMSE(i+1) = RMSEfis(FIS,Inp,Out); %Формую масив з RMSE, які обчислив на кожні
ітерації
end
Min_RMSE = flip(Min_RMSE);
[Min_RMSE_Value,Min_RMSE_Ind] = min(Min_RMSE(Line)); %Точність та індекс найкращьої
нечіткої бази знань
Min_Ind = [flip(Min_Ind) setdiff(Line_long,Min_Ind)];
counter
end
```

ДОДАТОК Г

Функція адитивного алгоритму найближчого сусіда

```
function [Min_RMSE,Min_Ind,Min_RMSE_Value,Min_RMSE_Ind] = ...
    func_add_neighbor(FIS,Line,Line_long,Inp,Out)
Min_RMSE = [];
Min_Ind = [];
num_output_mf = length(FIS.Outputs.MembershipFunctions);
inc_ind = [];
counter = 0;
%Зануляю ваги всіх правил, тобто вимключаю їх з бази знань
for j=Line_long
    FIS.rule(j).weight=0;
end
%Викликаю цикл який буде тривати до кількості правил
for i=Line
    RMSE_min = 10000; % Зміна яка буде зберігати мінімальне RMSE

    %Формую індекси для перебору правил, з виключенням тих правил,
    %що вже потрапили до нечіткої бази знань
    for j=setdiff(Line_long, Min_Ind)
        counter = counter + 1;
```

```
        FIS.rule(j).weight=1; %По черзі активую ваги правил
        RMSE = RMSEfis(FIS,Inp,Out); %Розраховую чергове RMSE
        %Знаходжу мінімальне RMSE та його індекс
        if RMSE_min > RMSE
            RMSE_min = RMSE;
            min_ind = j; % Зміна яка буде зберігати індекс мінімального RMSE
        end
        FIS.rule(j).weight=0; %Зануляю вагу чергового правила
    end

    FIS.rule(min_ind).weight=1;

    Min_RMSE = [Min_RMSE RMSE_min]; %Формую масив з RMSE, які обчислив на кожні
ітерації
    Min_Ind = [Min_Ind min_ind]; %Формую черговість індексів найкращих правил

    if rem(min_ind,num_output_mf) == 1
        inc_ind = [inc_ind
((min_ind+2):((idivide(int8(min_ind),int8(num_output_mf))+1)*num_output_mf))];
    elseif rem(min_ind,num_output_mf) == 0
        inc_ind = [inc_ind ((min_ind-2):-1:(min_ind/num_output_mf-
1)*num_output_mf+1)];
    else
        inc_ind = [inc_ind
((min_ind+2):((idivide(int8(min_ind),int8(num_output_mf))+1)*num_output_mf))];
        inc_ind = [inc_ind ((min_ind-2):-
1:((idivide(int8(min_ind),int8(num_output_mf))*num_output_mf+1)))];
    end
end
[Min_RMSE_Value,Min_RMSE_Ind] = min(Min_RMSE(Line)); %Точність та індекс найкращьої
нечіткої бази знань
counter
end
```

ДОДАТОК Д

Функція адитивного алгоритму бази знань з суперечливими правилами, яка в ході відбору виходить без суперечливих правил

```
function [Min_RMSE,Min_Ind,Min_RMSE_Value,Min_RMSE_Ind] = ...
    func_add_No_conf(FIS,Line,Line_long,Nrules,Inp,Out)
Min_RMSE = zeros([1,Nrules]);
Min_Ind = [];
closed_ind = [];
counter = 0;
%Зануляю ваги всіх правил, тобто вимключаю їх з бази знань
for j=Line_long
    FIS.rule(j).weight=0;
end
%Викликаю цикл який буде тривати до кількості правил
for i=Line
    RMSE_min = 10000; % Зміна яка буде зберігати мінімальне RMSE

    %Формую індекси для перебору правил, з виключенням тих правил,
    %що вже потрапили до нечіткої бази знань
    for j=setdiff(Line_long, closed_ind)
        counter = counter + 1;
        FIS.rule(j).weight=1; %По черзі активую ваги правил
        RMSE = RMSEfis(FIS,Inp,Out); %Розраховую чергове RMSE
        %Знаходжу мінімальне RMSE та його індекс
        if RMSE_min > RMSE
            RMSE_min = RMSE;
            min_ind = j; % Зміна яка буде зберігати індекс мінімального RMSE
        end
        FIS.rule(j).weight=0; %Зануляю вагу чергового правила
```

```
    end

    FIS.rule(min_ind).weight=1;

    Min_RMSE(i) = RMSE_min; %Формую масив з RMSE, які обчислив на кожні ітерації
    Min_Ind = [Min_Ind min_ind]; %Формую черговість індексів найкращих правил

    %Дізнаюсь словесну постановку найкращого правила та розбиваю її на частини
    name = split(FIS.Rules(min_ind).description);
    %Оголошую цикл для заблокування інших правил з такими ж правилами по x1 та x2
    for j=Line_long
        name_other = split(FIS.Rules(j).description);
        %Якщо словесні постановки по x1 та x2 від найкращого правила
        %співпадають з j, то запамятовуємо цей індекс, надалі він буде
        %заблокований
        if name_other(1) == name(1) && name_other(3) == name(3)
            closed_ind = [closed_ind j];
        end
    end
end
[Min_RMSE_Value,Min_RMSE_Ind] = min(Min_RMSE); %Точність та індекс найкращьої нечіткої
бази знань
counter
end
```

ДОДАТОК Е

Функція класичного алгоритму Ванга-Менделя

```
function [ FIS ] = WM( train,FIS )

Rulebase =zeros(size(train));
degree =zeros(size(train));

for i=1:size(train,2)
    mv=fsloc(train(:,i)',i,FIS);
    [degree(:,i), Rulebase(:,i)] =max(mv,[],2);
end

UniqueRules=unique(Rulebase(:,1:end-1),'rows');
d=prod(degree,2);

CompactRuleBase=zeros(size(UniqueRules,1),size(train,2));
for i=1:size(UniqueRules,1)
    tf=ismember(Rulebase(:,1:end-1),UniqueRules(i,:),'rows');
    p=tf.*d;
    [~, I]=max(p);
    CompactRuleBase(i,:)=Rulebase(I,:);
end

ruleList=cat(2,CompactRuleBase,ones(size(CompactRuleBase,1),2));

FIS=setfis(FIS,'rulelist',ruleList);
end
```

ДОДАТОК Ж

Функція модифікованого алгоритму Ванга-Менделя «сума голосів»

```
function [ FIS ] = WM_sum( train, FIS )
Rulebase =zeros(size(train));
degree =zeros(size(train));
```

```
for i=1:size(train,2)
    mv=fsloc(train(:,i)',i,FIS);
    [degree(:,i), Rulebase(:,i)] =max(mv,[],2);
end

UniqueRules = unique(Rulebase(:,1:end-1),'rows');
d=prod(degree,2);

UniqueRules_long=unique(Rulebase(:,1:end),'rows');
degree_long = zeros(size(UniqueRules_long,1),1);


for i=1:size(UniqueRules_long,1)
    tf=ismember(Rulebase(:,:),UniqueRules_long(i,:),'rows');
    degree_long(i) = sum(tf.*d);
end

CompactRuleBase=zeros(size(UniqueRules,1),size(train,2));
for i=1:size(UniqueRules,1)
    tf=ismember(UniqueRules_long(:,1:end-1),UniqueRules(i,:),'rows');
    p=tf.*degree_long;
    [~, I]=max(p);
    CompactRuleBase(i,:)=UniqueRules_long(I,:);
end
end
```

ДОДАТОК К

Функція модифікованого алгоритму Ванга-Менделя «головний конкурент»

```
function [ FIS ] = WM_competitor( train, FIS )
Rulebase =zeros(size(train));
degree =zeros(size(train));
for i=1:size(train,2)
    mv = fsloc(train(:,i)',i,FIS);
    [degree(:,i), Rulebase(:,i)] = max(mv,[],2);
end

Rulebase_competitor =zeros(size(train));
degree_competitor =zeros(size(train));
for i=1:size(train,2)
    mv = fsloc(train(:,i)',i,FIS);
    if i == 3
        for j=1:size(train,1)
            mv(j,Rulebase(j,3)) = 0;
        end
    end
    [degree_competitor(:,i), Rulebase_competitor(:,i)] =max(mv,[],2);
end

UniqueRules=unique(Rulebase(:,1:end-1),'rows');
d = prod(degree,2);
d_competitor = prod(degree_competitor,2);


CompactRuleBase=zeros(size(UniqueRules*2,1),size(train,2));
for i=1:size(UniqueRules,1)
    tf=ismember(Rulebase(:,1:end-1),UniqueRules(i,:),'rows');
    p=tf.*d;
    [~, I]=max(p);
    CompactRuleBase(i,:)=Rulebase(I,:);
end

for i=1:size(UniqueRules,1)
```

```
    tf=ismember(Rulebase_competitor(:,1:end-1),UniqueRules(i,:),'rows');
    p=tf.*d_competitor;
    [~, I]=max(p);
    CompactRuleBase(i+size(UniqueRules,1),:)=Rulebase_competitor(I,:);
end
Unique_CompactRuleBase=unique(CompactRuleBase,'rows');

ruleList=cat(2,Unique_CompactRuleBase,ones(size(Unique_CompactRuleBase,1),2));

FIS=setfis(FIS,'rulelist',ruleList);
end
```

ДОДАТОК Л

Функції візуалізації крих навчання, та поверхонь залежностей

```
function graph_1_line(Line,Min_RMSE_arr,Min_RMSE_Ind,Min_RMSE_Value,name_alg,type_FKB)
    figure('Position', [100 100 600 400]);
    plot(Line,Min_RMSE_arr,'k o',"LineStyle",'-','MarkerSize',8,...
    'MarkerEdgeColor','k','MarkerFaceColor',[1.0,0.0,0.0]);
    hold on
    plot(Min_RMSE_Ind,Min_RMSE_Value,'k s','MarkerSize',15);
    legend('RMSE_m_i_n','Best FKB');
    xlabel('Кількість правил')
    ylabel('RMSE')
    grid on
    title("Залежність RMSE від кількості найкращих правил, "+name_alg+"," +...
        newline + "бази знань, "+type_FKB+" RMSE(best)=" + num2str(Min_RMSE_Value,3));
end

function
graph_2_line(Line,Min_RMSE_arr_1,Min_RMSE_Ind_1,Min_RMSE_Value_1,name_alg_1,...
    type_FKB,Min_RMSE_arr_2,Min_RMSE_Ind_2,Min_RMSE_Value_2,name_alg_2)
figure;
plot(Line,Min_RMSE_arr_1,'k o',"LineStyle",'-','MarkerSize',8,...
    'MarkerEdgeColor','k','MarkerFaceColor',[1.0,0.0,0.0]);
hold on
plot(Min_RMSE_Ind_1,Min_RMSE_Value_1,'k o','MarkerSize',15);
hold on
plot(Line,Min_RMSE_arr_2,'k s',"LineStyle",'-','MarkerSize',8,...
    'MarkerEdgeColor','k','MarkerFaceColor',[0.0,0.0,1.0]);
hold on
plot(Min_RMSE_Ind_2,Min_RMSE_Value_2,'k s','MarkerSize',15);
legend(name_alg_1,strcat('Best FKB_R_M_S_E = ',num2str(Min_RMSE_Value_1,3)),...
    name_alg_2,strcat('Best FKB_R_M_S_E = ',num2str(Min_RMSE_Value_2,3)));
xlabel('Кількість правил')
ylabel('RMSE')
title("Залежність RMSE від кількості найкращих правил,"+...
    newline + "бази знань, "+type_FKB)
grid on
end

function
graph_3_line(Line,Min_RMSE_arr_1,Min_RMSE_Ind_1,Min_RMSE_Value_1,name_alg_1,...
    Min_RMSE_arr_2,Min_RMSE_Ind_2,Min_RMSE_Value_2,name_alg_2,...
    Min_RMSE_arr_3,Min_RMSE_Ind_3,Min_RMSE_Value_3,name_alg_3,type_FKB)
figure;
plot(Line,Min_RMSE_arr_1,'k o',"LineStyle",'-','MarkerSize',8,...
    'MarkerEdgeColor','k','MarkerFaceColor',[1.0,0.0,0.0]);
hold on
plot(Min_RMSE_Ind_1,Min_RMSE_Value_1,'k o','MarkerSize',15);
hold on
plot(Line,Min_RMSE_arr_2,'k ^',"LineStyle",'-','MarkerSize',8,...
    'MarkerEdgeColor','k','MarkerFaceColor',[0.0,1.0,0.0]);
hold on
plot(Min_RMSE_Ind_2,Min_RMSE_Value_2,'k ^','MarkerSize',15);
hold on
plot(Line,Min_RMSE_arr_3(Line),'k s',"LineStyle",'-','MarkerSize',8,...
    'MarkerEdgeColor','k','MarkerFaceColor',[0.0,0.0,1.0]);
hold on
```

```
plot(Min_RMSE_Ind_3,Min_RMSE_Value_3,'k s','MarkerSize',15);
legend(name_alg_1,strcat('Best FKB_R_M_S_E = ',num2str(Min_RMSE_Value_1,3)),...
    name_alg_2,strcat('Best FKB_R_M_S_E = ',num2str(Min_RMSE_Value_2,3)),...
    name_alg_3,strcat('Best FKB_R_M_S_E = ',num2str(Min_RMSE_Value_3,3)));
xlabel('Кількість правил')
ylabel('RMSE')
title("Залежність RMSE від кількості найкращих "+newline+"правил "+type_FKB)
grid on
end

function graph_surf(Line,Lim_Z,Num_best_rules,Min_Ind,FIS,name_alg,type_FKB,X1,X2,Y)
    for i=Line
        FIS.rule(i).weight=0;
    end
    for i=1:Num_best_rules
        FIS.rule(Min_Ind(i)).weight=1;
    end
    figure;
    plot3(X1,X2,Y,'go','LineStyle','none','MarkerSize',5,...
        'LineWidth',1,'MarkerEdgeColor','k','MarkerFaceColor',[0.0,0.0,1.0]);
    title("Поверхня найкращої бази знань, "+type_FKB+","+newline+...
        "отримана, "+name_alg);
    hold on
    gensurf(FIS);
    zlim(minmax(Lim_Z));
    colormap('summer(20)')
    grid on
    legend("Тестові значення","Поверхня бази знань");
    alpha(.6)
end
```

Резнік Роман Юрійович
Прізвище, ім'я по батькові

Факультет інформаційних та прикладних технологій
Факультет

122 «Комп'ютерні науки»
Шифр і назва спеціальності

Комп'ютерні технології обробки даних (DataScience)
Освітня програма

ДЕКЛАРАЦІЯ АКАДЕМІЧНОЇ ДОБРОЧЕСНОСТІ

Усвідомлюючи свою відповідальність за надання неправдивої інформації, стверджую, що подана кваліфікаційна (магістерська) робота на тему:

« Ідентифікація багатофакторних залежностей на основі нечітких баз знань з суперечливими правилами» є написаною мною особисто.

Одночасно заявляю, що ця робота:

- не передавалась іншим особам і подається до захисту вперше;
- не порушує авторських та суміжних прав, закріплених статтями 21-25 Закону України «Про авторське право та суміжні права»;
- не отримувалась іншими особами, а також дані та інформація не отримувалась у недозволений спосіб.

Я усвідомлюю, що у разі порушення цього порядку моя кваліфікаційна робота буде відхилена без права її захисту, або під час захисту за неї буде поставлена оцінка «незадовільно».

12.12.2022 _____
(дата) (підпис здобувача освіти)