

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ АЗЕРБАЙДЖАНСКОЙ РЕСПУБЛИКИ  
АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ**

**МАГИСТЕРСКИЙ ЦЕНТР**

*На правах рукописи*

**Алиев Кямран Эльшад оглы**

**По теме: “Методы и средства моделирования  
слабоструктурированных информационных процессов”**

**МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ**

<b>Название и шифр специальности</b>	<b>060509</b>	<b>Компьютерные науки</b>
<b>Специализация</b>		<b>Экономические информационные системы</b>
<b>Научный руководитель:</b>		<b>Руководитель магистерской программы:</b>
<b>к. ф.-м.н., доц. Байрамов Х.М</b>		<b>к. ф.-м.н., доц. Байрамов Х.М</b>
<b>Заведующий кафедрой:</b>		<b>к. ф.-м.н., доц. Байрамов Х.М</b>

**Баку – 2018  
ОГЛАВЛЕНИЕ**

<b>ВВЕДЕНИЕ</b> .....	3
<b>ГЛАВА I. ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ И СПОСОБЫ ИХ МОДЕЛИРОВАНИЯ</b> .....	6
1.1. Понятие информации. Ее роль в информационных процессах.....	6
1.2. Понятие «информационные процессы».....	10
1.3. Понятие «модель». Методология моделирования.....	12
<b>ГЛАВА II. НЕЙРО-НЕЧЕТКИЕ МЕТОДЫ ФОРМАЛИЗАЦИИ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ</b> .....	22
2.1. Нечеткая логика и основные элементы нечеткой математики.....	22
2.2. Механизм нечеткого вывода.....	27
2.3. Нейронные сети.....	34
2.4. Нейро-сетевое моделирование информационных процессов.....	40
<b>ГЛАВА III. НЕЙРО-НЕЧЕТКИЕ МЕТОДЫ И СРЕДСТВА МОДЕЛИРОВАНИЯ СЛАБОСТРУКТУРИРОВАННЫХ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ</b> .....	46
3.1. Нейро-нечеткое моделирование.....	46
3.2. Моделирование слабоструктурированных процессов методами и средствами ANFIS.....	55
<b>ВЫВОДЫ И ПРЕДЛОЖЕНИЯ</b> .....	72
<b>СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ</b> .....	73

**Актуальность темы:** В последнее время информационные процессы играют важную роль в любой сфере деятельности. Различные действия, которые производятся с информацией, называются информационными процессами. Любой информационный процесс, это последовательность операций во времени при проектировании, производстве, обслуживании и управлении. При разработке какого-либо проекта, информационные процессы, сопровождаются этапами процессов: от возникновения той или иной идеи проектирования проекта, определения конфигурации этого проекта и вплоть до его реализации, анализа результатов, и его использования.

Информационные процессы делятся на структурированные, слабоструктурированные, неструктурированные. В последнее время широкое распространение получили слабоструктурированные информационные процессы.

Слабоструктурированными информационными процессами, называются процессы, которые находятся в промежутке между структурированными и неструктурированными процессами. Слабоструктурированные процессы имеют большую гибкость, чем структурированные, так как они выполняются более неопределенно и им приходится иметь дело с изменениями и исключениями. А, в отличие от неструктурированных, они могут описываться достаточно формализовано.

Слабоструктурированные процессы - процессы, в которых доминируют качественные, плохо определенные факторы. Слабоструктурированные задачи сочетают количественные и качественные зависимости, причем малоизвестные и неопределенные стороны задачи имеют тенденцию доминировать. Особенностью слабоструктурированных информационных процессов, является неопределенность входных и выходных данных, а также изменение экземпляров процесса.

На этапе моделирования и автоматизации слабоструктурированных процессов практически невозможно заранее принять решение о компонентах процесса и их действиях. Таким образом, невозможно создать детальное протекания процессов, на основании которых можно было бы обрести контроль его выполнение.

Для решения широкого круга задач, таких как прогнозирование, поддержка принятия решений, управления, описания и т.д., связанных со слабоструктурированными процессами, используются нейро-нечеткие методы и средства моделирования слабоструктурированных и плохо формализованных процессов.

**Цели и задачи исследования:** Целью этой диссертационной работы является исследование методов и средств моделирования слабоструктурированных информационных процессов протекающих сегодня в любой сфере деятельности. Основной задачей является:

- выделение основных категорий существующих методов анализа слабоструктурированных процессов,
- сформулировать контекст применения этих самых методов, для конкретики требований, которые должны решить эту задачу,
- оценить эффективность применения рассмотренных методов, возможность комбинирования друг с другом, выделить особенности их реализации в контексте,
- дать оценку преимуществ и недостатков их применения.

Применить эту модель для описания реального слабоструктурированного процесса и расширить модель инструментом оценки.

**Объект и предмет исследования:** Объектом исследования является слабоструктурированные процессы протекающие в сферах деятельности. Предметом исследования является механизм моделирования и применения моделей для более выгодных результатов.

**Теоретические и методологические основы исследования:**

Составляют труды зарубежных ученых в сфере нечеткой логики и нейронных сетей.

**Научная новизна:** Заключается в рассмотрении нейро-нечетких методов и средств моделирования и применения их к слабоструктурированным процессам в MATLAB (ANFIS redactor).  
Оценивание эффективности этой модели для моделирования.

**Информационная база исследования:** Состоит из обработанных материалов, которые включали методы и средства моделирования на основе нейро-нечеткой модели к слабоструктурированным процессам.

**Объем и структура исследования:** Диссертационная работа состоит из введения, 3 глав в сумме содержащих 9 разделов, вывода и списка используемой литературы.

1-ая глава говорит о простейших понятиях информации, информационного процесса и об обобщенном понятии методов моделирования, которые можно применить к информационным процессам.

2-ая глава описывает нечеткую логику и его механизм вывода, нейронные сети и их модели для моделирования.

3-я глава описывает применение гибридного метода нечеткой логики и нейронных сетей для моделирования слабоструктурированных процессов, рассмотрение нейро-нечетких моделей, применение этих моделей в реальных процессах.

# ГЛАВА I. ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ И СПОСОБЫ ИХ МОДЕЛИРОВАНИЯ

## 1.1 Понятие информации. Ее роль в информационных процессах

В нашем сегодняшнем мире информация играет очень важную роль в любой сфере деятельности. Информация (перевод с лат. *informatio* представление, описание) является ключом в нынешней науке, и находится наряду со следующими понятиями подобно "энергия" и "материя". Несмотря на то что есть много понятий "информации", можно выделить несколько основных интерпретаций понятия "информация": научная интерпретация, абстрактная интерпретация, конкретная интерпретация.

В научной трактовке информация есть исходная общенаучная категория, которая отражает как структурность объекта, так, и методы ее постижения, которая не сводится к более легким положениям. В абстрактной интерпретации, она рассматривается как некий порядок символьных значений, которые обладают неким смысловым понятием для выполняющего принятие или передачу, как обобщенно, так и частности. В конкретной интерпретации просматриваются определенные исполняющие с применением специфических системных команд и семантического языка. К примеру, для животных, людей это виденье, звуки; со стороны технологий информация это «0 и 1» и т.д. Есть некоторые концепции парадигм информации.

1-ое понятие это концепция Шеннона, которая отражает количественное направление, определяющая информацию как некую грань неточности – энтропию явления.

Во 2-ой ее рассматривают как свойство вещества (ее атрибут). Такая концепция появилась благодаря эволюции кибернетики и основывается на том, что информация содержит всякие сообщения, которые воспринимаются и людьми, и приборами.

И наконец 3-я формируется на логико-семантическом направлении, в этой концепции информация излагается как знание, притом такая часть знания, которая применяется с целью ориентации, с целью активных действий, с целью контроля и самоконтроля. Другими изъяснениями, информация нужная, важная, действенная, "используемая" часть знаний. (Концепция В.Г. Афанасьева.)

В современные дни слово "информация" обладает углубленным содержанием и множественными взглядами. Информация также обладает свойствами, и ее можно разделять на виды.

Информация обладает нижеперечисленными свойствами:

- полезность;
- полнота и понятность;
- достоверность;
- актуальность;
- доступность;
- защищенность;
- объективность.

Полезность информации это свойство которое говорит насколько она является ценной в той или иной сфере деятельности.

Полнота, понятность свойство, определяющее, представляется ли информация полной, т.е. dokonченной по смыслу, и ясна ли она получающему ее.

Достоверность свойство информации, определяющее ее истинность, насколько она достоверная для той сферы, в которой она будет применяться.

Актуальность и объективность свойство, определяющее насколько полученная информация, будет пригодной для настоящего времени, не утратила ли она свою пригодность.

Доступность и защищенность информации определяют насколько информация секретна и доступна, и обладает ли она защитой от несанкционированного доступа. Информацию можно также разделить на нижеперечисленные виды, от способа принятия этой информации:

- 1) Визуальная информация, которая постигается благодаря зрению.
- 2) Аудиальная информация воспринимаемая благодаря слуху.
- 3) Тактильная информация постигается благодаря осязанию объектов, предметов и т.д.
- 4) Вкусовая информация постигается благодаря вкусовым рецепторам.
- 5) Обонятельная информация воспринимаемая в виде запахов.

Все эти виды информации исходят из способа принятия информации людьми. Исследования этого наукой доказало, что с помощью зрения и слуха человек постигает около 90% информации в окружающей его среде, а остальные 10% информации воспринимается благодаря тактильным, вкусовым и обонятельным качествам. Человек чтобы получить информацию, являющейся недоступной ему, благодаря качествам изложенным выше, изобретает различные приспособления и приборы: приспособления измерения температуры и давления, приборы наблюдения, мониторинга и т.д. Человек принимает информационные сигналы благодаря органам чувств, а в приборах эту роль играют датчики.

Поэтому информация по способу принятия делится также на последующие виды: аналоговая и цифровая (рисунок 1.1).

При поступлении информации посредством органов чувств, она принимается непрерывно, и этот вид называется аналоговой. А в приспособлениях, роль принятия информации играют микро-датчики, в отличии от органов чувств они принимают ее дискретно.

Но как и везде, и среди приспособления есть исключения, некоторые из которых также имеют дело с непрерывной информацией (телек, телефон и т.д.).

Также непрерывную информацию есть возможность превратить в цифровую, и цифровую в непрерывную, для этого существуют устройства двух типов «Цифро-аналоговый преобразователь» и «Аналого-цифровой преобразователь».

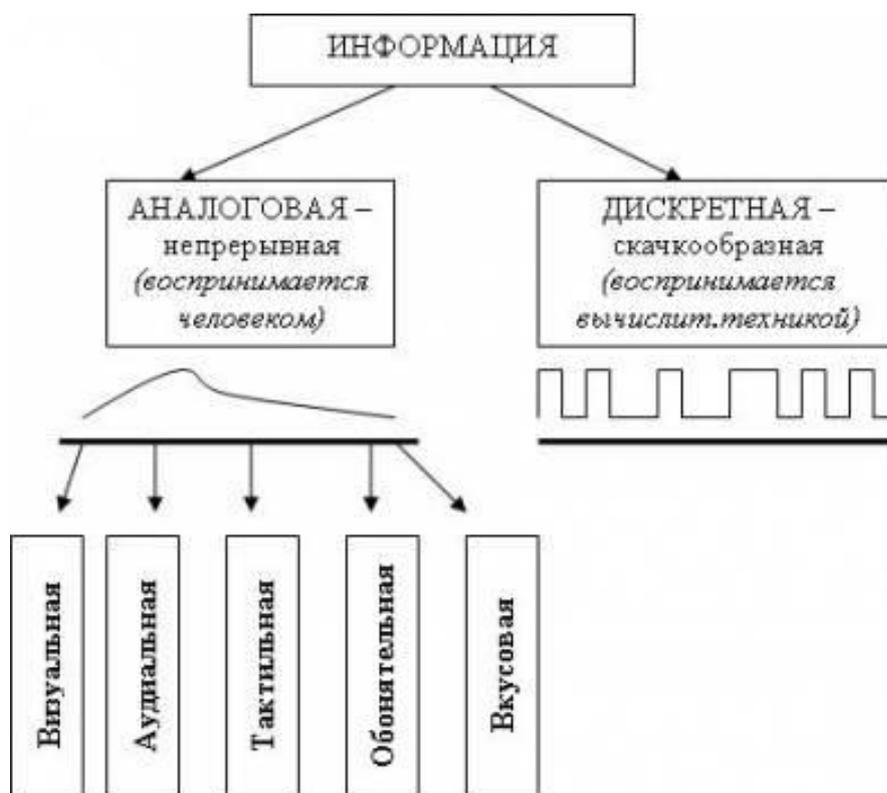


Рисунок 1.1. Аналоговая и дискретная информация.

Значение информации в информационных процессах – есть основа. В любой сфере деятельности человек использует информацию, проводя ее поиск, хранение, передачу, обработку и вывод этой информации для произведения различных видов задач в той или иной сфере деятельности. Мы сказали, что информация поступает к человеку путем органов чувств и приборов.

С целью выполнения хранения и обработки информации человек использует мозг, в котором выполняются операции благодаря нейронам мозга. Человек выполняя такие операции, например написание чего-либо, разговаривая с кем-либо, читая статьи и т.д., работает с информацией. Операции с ней выполняются не только в человеческой сфере, но в окружающей среде. Все что окружает нас имеет связь с информацией, с ее переработкой.

Все операции которые производятся с ней в окружающей среде, и в сфере функционирования человека, можно собрать и назвать термином «информационные процессы».

## **1.2 Понятие информационного процесса (ИП)**

Мы сказали, что ИП – это процесс поиска, введения, обработки, передачи, и представления информации, для выполнения задач посредством этой информации (ее также можно назвать знаниями). Информация – важные факты, полученные из данных. Знание это информация, которая была протестирована и представлена в виде законов, теорий. ИП это цикл создания информации и ее сохранения в виде новых данных.

В информационный процесс входят следующие действия с данными:

Сбор данных это накопление данных, чтобы обеспечить его полноту для принятия дальнейших решений;

Формирование данных данные, полученные из разных источников, преобразуются в однородность;

Фильтрация данных удаляет «лишние» данные, которые не нужны для принятия решения;

Сортировка данных упорядочивает данные по определенному признаку;

Группирование данных это объединение данных по определенным характеристикам;

Архивирование данных организует хранение данных в удобной и легкодоступной форме, обычно в более экономичном формате;

Защита данных это набор мер, направленных на предотвращение потери, модификации или воспроизведения данных.

Информационный процесс можно разделить на несколько групп:

- поиск;
- хранение;
- обработка;
- передача;

➤ представление.

Поиск процесс нахождения информации, из источников хранения (книги, статьи, журналы и т.п.), для выполнения операций и ее применения в будущем.

Хранение процесс который нужен с целью чтобы сохранить информацию, затем применить ее в будущем. Средствами хранения информации могут выступать: книги, журналы, электронные носители, статьи в интернете и т.д.

Обработка процесс выполнения алгоритмических действий с информацией, для нахождения ищущейся или новой информации.

Передача информации это обмен информацией между двумя или несколькими источниками. При передаче обязательно существует отдающий и принимающий информацию, независимо от того является ли это человек или что-то.

Представление процесс итога решения информации, в том виде в котором нужно, для использования. Представление может быть в виде картинок, таблиц, графов, уравнений, формул и т.д.

ИП по структуризации подразделяются на:

- структурированные информационные процессы;
- слабоструктурированные информационные процессы;
- неструктурированные информационные процессы.

Структурированные ИП это такие процессы которые формализованы с помощью количественных данных, количественной информации. Структурированные ИП легко поддаются программированию и математико-алгоритмическим действиям.

Неструктурированные ИП это полная противоположность к структурированным, строятся эти ИП на основе качественных данных, и очень сложны с целью употребления в современное время.

Слабоструктурированные ИП обладают и количественными, так и качественными данными, но в большей степени процессы протекают с

качественными данными, и имеют дело с плохо формализованными данными. Эти информационные процессы плохо поддаются программированию, и имеют дело с исключениями и изменениями данных во времени.

Слабоструктурированные ИП находятся между структурированными и неструктурированными, являются очень гибкими по отношению к структурированным.

Слабоструктурированные процессы сегодня широко применяются в банковской сфере, где потоки данных которые втекают в банк, являются качественными. Моделирование этих слабоструктурированных процессов является трудоемким процессом, имеет дело с неопределенностью из-за представления именно этих качественных данных.

### **1.3 Понятие «модель». Методология моделирования.**

Первым делом не приступая к методам и средствам моделирования, нужно понимать, что из себя в первую очередь представляют термины “model” и “modelling”. Модель представление подобия любого предмета в простом виде, который обладает свойствами и характеристиками оригинального предмета.

Моделирование процесс, с применением подобия любого предмета, с исследованиями свойств, связей, качеств и параметров оригинального предмета и применением их при принятии решений и прогноза состояний в будущем.

Существует много методов и средств моделирования, которые широко применяются в различных сферах деятельности. Modelling оригинальных предметов, событий происходит, для того чтобы прогнозировать их действия в будущем. Поговорим о классификациях моделирования.

### Детерминированное моделирование.

Детерминированные модели строятся на основе математических закономерностей, и с помощью этих закономерностей результат оказывается более точным. Детерминированное построение (modeling) высвечивает те события, которые не имеют случайных воздействий со стороны.

При выводе результата исследования процессов, с помощью этого моделирования, выводится единственно точный результат (рисунок 1.2).

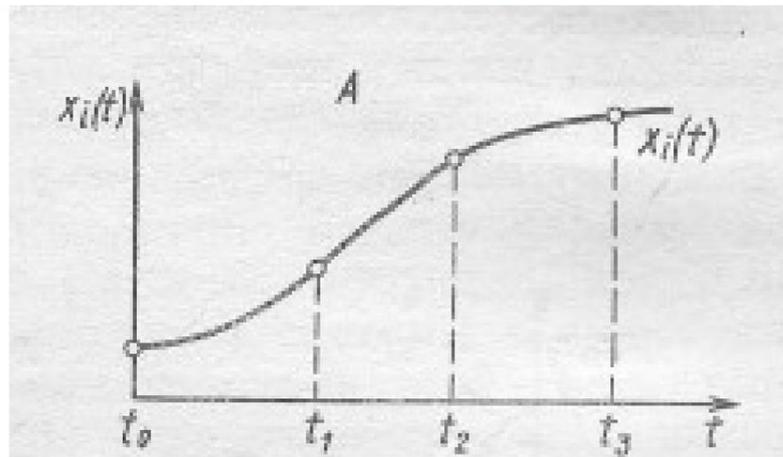


Рисунок 1.2. График детерминированного построения (modeling).

### Стохастическое моделирование

Стохастическая модель является сложной при конструировании, состоящих из нескольких шагов. Приемы строятся на базе математической статистики, и выдвигает вероятности операций. Благодаря стохастической модели выводятся несколько результатов исследуемого предмета, и происходит оценка среднего значения показателей, то есть множество однородных результатов (рисунок 1.3).

Стохастическое моделирование можно считать, как дополнение к детерминированному моделированию, используется в тех случаях где есть неопределенности, и нельзя прибегнуть к детерминированному моделированию.

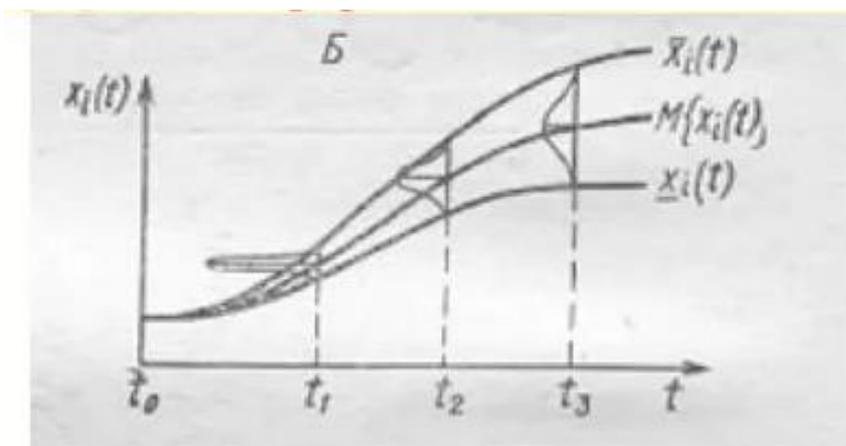


Рисунок 1.3. График стохастического моделирования.

### Статическое и динамическое моделирование.

Статическое построение (modeling) отображает показатели событий в определенный момент  $t$  или не учитывает временные факторы. Отображение показателей в определенный момент  $t$ , называется статистическим срезом, которое показано на рисунке 1.4.

Динамическое построение (modeling) отображает показатели в полное время ( $t$ ), то есть весь цикл исследуемого события.



Рисунок 1.4. Статическое и динамическое моделирование.

### Дискретное, непрерывное и дискретно-непрерывное моделирование.

Дискретное построение (modeling) отображает показатели процесса в дискретный момент  $t$ , к примеру в промежутке  $t_0 \dots t_1$ .



Рисунок 1.5. Дискретное моделирование.

Непрерывное построение (modeling) отображает показатели процесса, которые меняются непрерывно с течением  $t$ . Например состояние процесса в момент времени  $t_1, t_2 \dots t_n$ .

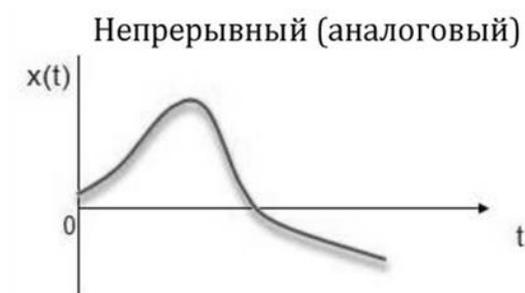


Рисунок 1.6. Непрерывное моделирование.

Чтобы представить показатели процесса какого-либо предмета (процесса), как дискретно-событийно, так и непрерывно, применяется дискретно-непрерывное построение (modeling).

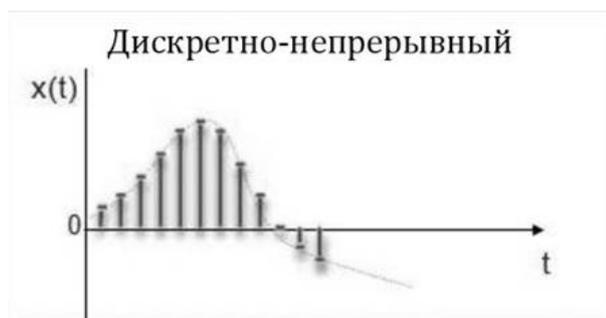


Рисунок 1.7. Дискретно-непрерывное моделирование.

Мысленное построение (modeling) называется таким моделированием предметов и событий, которые не возможно реализовать на практике в естественных условиях с протеканием времени. Благодаря этому моделированию можно исследовать процессы на микроуровне, которые не поддаются реализации в физически-экспериментальных условиях.

Мысленное построение делится на:

- наглядное;
- символическое;
- математическое.

Наглядное моделирование иллюстрирует процессы, происходящие внутри процессов, предметов, состояний. Примерами наглядного построения, являются диаграммы, схематические картины, макеты, диаграммы и т.д.

Символическое построение (modeling) создает процессы предмета протекающие в искусственном виде, действия и свойства которого определяются знаками и символами, и схожи с реальным процессом. К примерам можно отнести языки программирования, уравнения и т.д.

Для отображения любого явления, предмета в виде математических соотношений или алгоритма, применяется математическое построение (modeling). Главной целью этого построения (modeling), является использование математических методологий для исследования событий, возникающих реально, как и в оригинале.



Рисунок 1.8. Создание математической модели.

Математическая модель представляет процессы, приближенные к более точным процессам оригинального (истинного) предмета. Эта модель может состоять из математических уравнений, включающих переменные, действия которого схожие, с действиями реального процесса.

Математическое моделирование процессов делится на:

- аналитическое;
- имитационное;
- комбинированное.

Простейший пример: уравнения **Лотки-Вольтерры**  
на основе уравнения логистического роста популяции:

$$\frac{dN}{dt} = rN \frac{K - N}{K}$$

Популяции хищника и жертвы

$$\frac{dN_1}{dt} = r_1 N_1 - p_1 N_1 N_2$$

$$\frac{dN_2}{dt} = p_2 N_1 N_2 - d_1 N_2$$

Конкурирующие популяции

$$\frac{dN_1}{dt} = r_1 N_1 \frac{K_1 - N_1 - \alpha_{12} N_2}{K_1}$$

$$\frac{dN_2}{dt} = r_2 N_2 \frac{K_2 - N_2 - \alpha_{21} N_1}{K_2}$$

Рисунок 1.9. Пример математического моделирования.

### Аналитическое моделирование.

Протекание процессов в системах, в аналитическом моделировании фиксируются наподобие функциональных зависимостей и логических выражений.

$$\text{MM} = \begin{cases} L_1(x_1, x_2, \dots, x_n, y_1) \leq b_1 \\ L_2(x_1, x_2, \dots, x_n, y_2) \geq b_2 \\ L_3(x_1, x_2, \dots, x_n, y_3) > b_3 \\ \dots \\ G(x_1, x_2, \dots, y_1, y_2, \dots) \rightarrow opt \end{cases}$$

Рисунок 1.10. Аналитическая математическая модель.

Где,  $x_1, x_2, \dots, x_n$  и  $y_1, y_2, \dots, y_m$  – параметры моделируемого объекта;  
 $b_1, b_2, \dots, b_k$  – ограничения, накладываемые на функционирование объекта окружающей средой;

$G(x_1, x_2, \dots, y_1, y_2, \dots)$  – целевая функция моделирования.

Анализ в аналитической модели протекает с такими методами, как:

1) аналитический, который используется для нахождения зависимостей в общем виде, между введенными свойствами и характеристиками.

2) численный метод, применяется, если не получается решить уравнение в общем виде, с нахождением числовых значений и вводом исходных значений.

3) качественный метод, применяется, если нельзя решить уравнение в определенном виде, чтобы обнаружить особенности вычислений.

### Имитационное моделирование.

Имитационное моделирование является методом построения прототипа, исследуемого процесса, который выполняет процессы с точностью как реальные. Благодаря этой модели можно следить за действиями системы во временном факторе. Плюсом этой модели является, его применение к более сложным задачам, в отличие от аналитической модели.

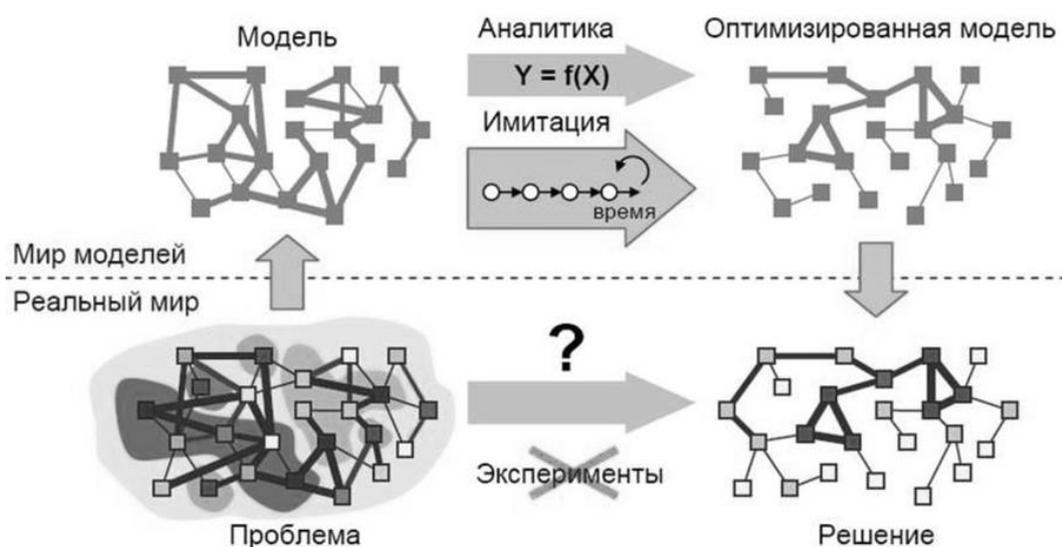


Рисунок 1.11. Пример имитационного моделирования.

В современное время имитационное построение можно назвать одним из эффективных способов экспериментального наблюдения реальных объектов и широко используется при познании сложных систем.

### **Комбинированное моделирование.**

Комбинированное построение (modeling) включает в себя две выше перечисленные модели. При проектировании этой модели производится расщепление процессов на под-процессы, в местах расщепления применяется аналитическое, а где нет расщепления имитационное.

Такая модель помогает исследовать системы, которые не возможно исследовать, только имитационным или аналитическим моделированием.

Реальное моделирование такой метод построения, который применяется для наблюдения процессов на самом предмете (original). Из за наблюдения на реальном предмете, возможности этого моделирования являются ограниченными.

Реальное моделирование делится на:

- натурное;
- физическое.

Натурное моделирование производит обработку значений, найденных после изучения реального предмета (явления), на основе теорий подобия. Физическое построение (modeling) воссоздает внешнюю среду, и изучает действия предмета (оригинала) или ее прототипа, независимо от временного фактора.

### **Кибернетическое моделирование.**

Кибернетическое построение (modeling) играет важную роль в моделировании, и в нем не существует схожести процессов к оригиналу, протекающих в модели.

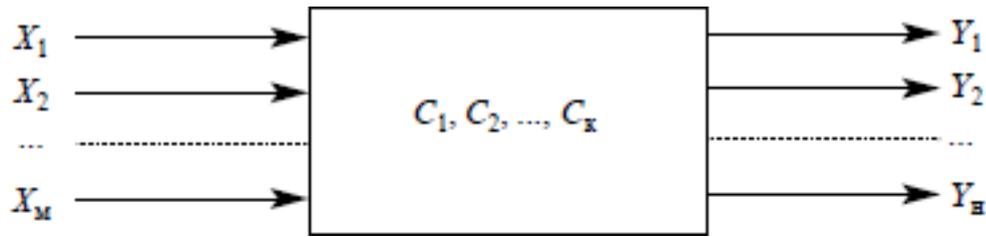


Рисунок 1.12. Кибернетическое моделирование.

Она создается на основе формальных отображений процессов, со связью с внешней средой. Связь с внешней средой обозначается понятиями “входа” и “выхода”.

$X_1, X_2, \dots, X_M$  – входы;

$Y_1, Y_2, \dots, Y_N$  – выходы;

$C_1, C_2, \dots, C_K$  – состояние.

На основе кибернетического моделирования лежит метод “черного ящика”, который рассматривает поведения реального объекта (выходной информации), с воздействиями внешней среды (входной информации), независимо от процессов протекающих внутри самого объекта.



Рисунок 1.13. Метод “черного ящика”.

### Когнитивное моделирование.

В настоящее время одна из развивающихся областей когнитивных наук является когнитивное моделирование, которое является междисциплинарным научным направлением, объединяющее теорию познания, когнитивной психологии, нейрофизиологии, теории когнитивной лингвистики и искусственного интеллекта. Когнитивное моделирование, помогает решить

ряд задач прогнозирования. Традиционно, когнитивная модель представляет собой когнитивные карты в виде ориентированного графа (орграфа)  $G = \langle V, E \rangle$ , где:

$V$  представляет собой набор понятий (концепций),

$V_i \in V, i=1, 2, \dots, k$  являются элементами системы исследования;

$E$  представляет собой множество дуг, а дуги  $e_{ij} \in E, i, j=1, 2, \dots, N$  отражают соотношение между понятиями.

Когнитивный карта позволяет анализировать как качественные, так и количественные данные, а общая цель когнитивного моделирования заключается в создании и тестирование гипотезы о функциональной структуре наблюдаемой ситуации. Итерации проводятся до определенного состояния причинно-следственных отношений, структурная схема которого позволит нам понять и анализировать поведение системы.

Рассмотрев основу методологии моделирования, можно сказать что все вышеперечисленные методы моделирования можно применять к информационным процессам. Но на сегодня существует много методов и гибридов для моделирования, которые будут рассмотрены в следующих главах.

## ГЛАВА II. НЕЙРО-НЕЧЕТКИЕ МЕТОДЫ ФОРМАЛИЗАЦИИ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ

### 2.1. Нечеткая логика и основные элементы нечеткой математики

За последние несколько лет наблюдался быстрый рост числа и разнообразия приложений нечеткой логики (FL). Методы FL использовались в приложениях для понимания изображений, таких как обнаружение краев, извлечение признаков, классификация и кластеризация. Нечеткая логика создает способность имитировать человеческий разум, эффективно использовать методы рассуждения, которые являются приближительными, а не точными. В традиционных жестких вычислениях, решения или действия основаны на точности, уверенности и силе.

Нечеткая логика начинается с понятия нечеткого множества. Нечеткое множество - это набор без четкой, четко определенной границы. Он может содержать элементы с частичной степенью принадлежности. Классическое множество - это контейнер, который полностью включает или полностью исключает любой данный элемент. Другая версия этого закона гласит: «Из любого предмета нужно либо утверждать, либо отрицать». То есть переход от «принадлежности к множеству» к «не принадлежащему множеству» является постепенным, и этот плавный переход характеризуется функцией принадлежности, которая дает гибкость множеству нечетких множеств при моделировании обычно используемых лингвистических выражений.

Функция принадлежности (MF) - это кривая, определяющая, как каждая точка в пространстве ввода отображается на значении принадлежности (или степени принадлежности) между 0 и 1. Входное пространство иногда называют универсумом дискурса (пространство рассуждений). Выходная ось представляет собой число, известное как значение принадлежности между 0 и 1. Кривая известна как функция принадлежности и часто дается обозначение  $\mu$ . Простейшие MF формируются с использованием прямых

линий. Из них наиболее простой является функция треугольной принадлежности, и она имеет имя функции `trimf`. Эта функция представляет собой ни что иное, как совокупность трех точек, образующих треугольник. Функция трапециевидной принадлежности, `trapmf`, имеет плоскую вершину и на самом деле представляет собой только усеченную треугольную кривую. Эти функции линейной принадлежности имеют преимущество простоты.

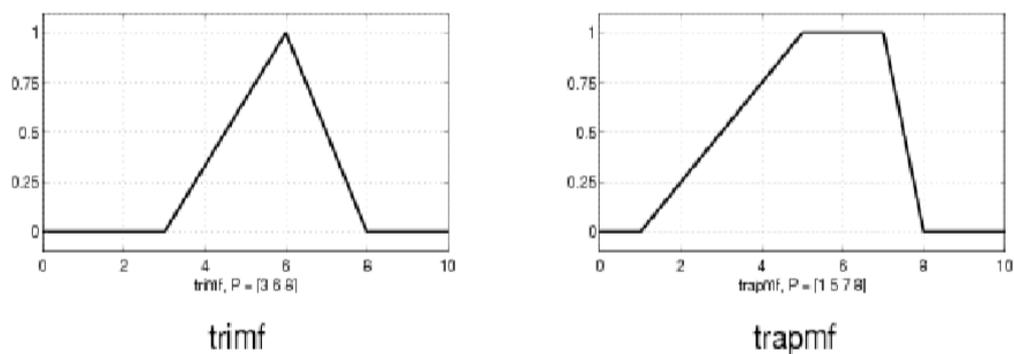


Рисунок 2.1. Треугольной и Трапециевидной MF.

На кривой распределения Гаусса построены две функции принадлежности: простая гауссова кривая и двухсторонний композит двух разных гауссовых кривых. Две функции являются `gaussmf` и `gauss2mf`. MF “обобщенного купола” определяется тремя параметрами и имеет имя функции `gbellmf`.

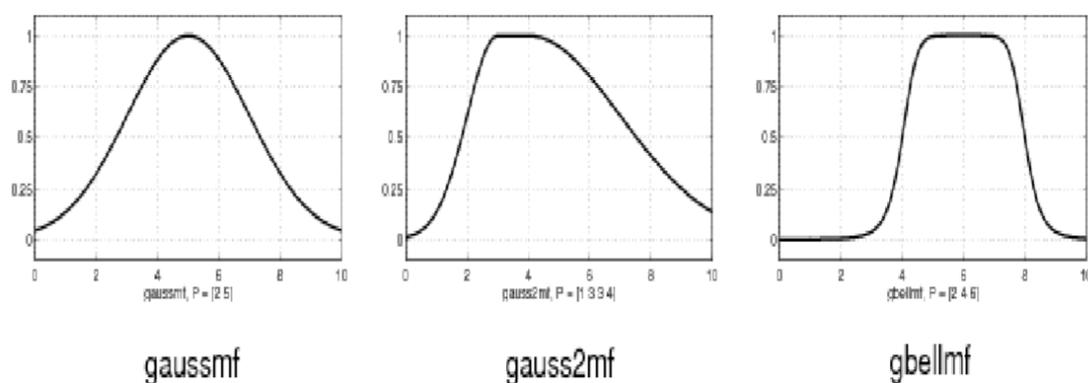


Рисунок 2.2. MF Гаусса и “обобщенного купола”.

Также мы можем определить сигмоидальную MF, которая либо открыта влево, либо вправо. Асимметричные и закрытые (т.е. не открытые слева или справа) MF могут быть синтезированы с использованием двух сигмоидальных функций, поэтому в дополнение к основному `sigmf`, мы также имеем разницу между двумя сигмоидальными функциями – `dsigmoid` и произведение двух сигмоидальных функций `psigmoid`.

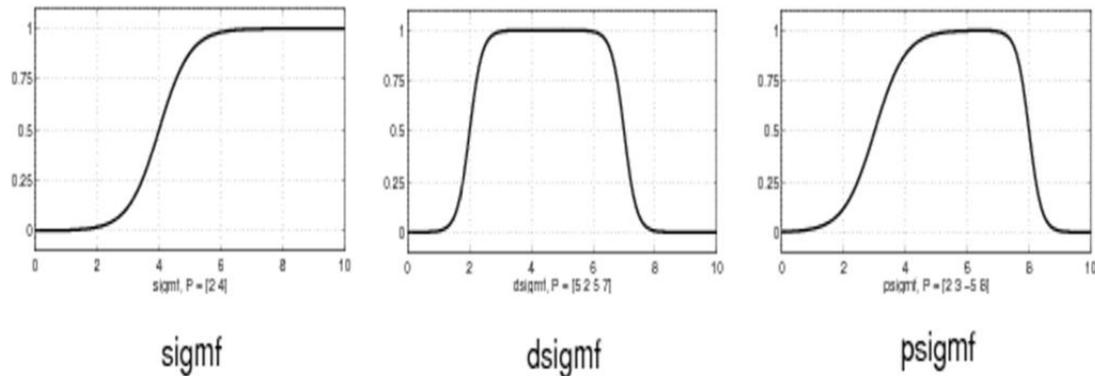


Рисунок 2.3. Сигмоидальные MF.

### Нечеткое правило *If-Then*.

Нечеткое правило *If-Then* принимает вид:

$$\text{If } x \text{ is } A, \text{ then } y \text{ is } B, \quad (2.1)$$

где  $A$  и  $B$  - лингвистические значения, определяемые fuzzy множествами в пространстве рассуждений  $X$  и  $Y$  соответственно. Часто « $x$  is  $A$ » называется антецедентом или предпосылкой, тогда как « $y$  is  $B$ » называется следствием или заключением. Примеры нечетких правил *if-then* в наших ежедневных лингвистических выражениях таковы: если дорога скользкая, вождение опасно. Прежде чем мы сможем использовать fuzzy правила *if-then* для моделирования и анализа системы, мы сначала должны формализовать то, что понимается под выражением *if  $x$  is  $A$ , then  $y$  is  $B$* , что иногда сокращается как  $A \rightarrow B$ . по существу, выражение описывает связь между двумя переменными  $x$  и  $y$ ; это предполагает, что нечеткое правило *if-then*

определяется как двоичное нечеткое отношение  $R$  на пространстве произведений  $X \times Y$ . Бинарное отношение  $R$  является расширением классического декартова произведения, причем каждый элемент  $(x, y) \in X \times Y$  связан с классом принадлежности, обозначаемым  $\mu_R(x, y)$ . Интерпретация правила *if-then* включает в себя различные части: сначала оценивает антецедент (который включает в себя fuzzifying вход и применение любых необходимых нечетких операторов), а второй применяет этот результат к последующему (известному как импликация). В случае двузначной или двоичной логики правила *if-then* не представляют большой трудности.

В большинстве приложений решение FL представляет собой перевод человеческого решения. В-третьих, FL может моделировать нелинейные функции произвольной сложности с требуемой степенью точности. FL - удобный способ отобразить входное пространство в выходное пространство. FL является одним из инструментов, используемых для моделирования многопользовательской многоуровневой системы.

В зависимости от структуры правил if-then можно выделить два основных типа нечетких моделей: модель Мамдани (или лингвистическая) и модель Такаги-Сугено.

### **Модель Мамдани.**

В этой модели антецедент (if – часть правила) и последующая (then – часть правила) являются нечеткими предложениями:

$$\mathcal{R}_i: \text{If } x \text{ is } A_i \text{ then } y \text{ is } B_i, \quad i = 1, 2, \dots, K \quad (2.2)$$

Здесь  $A_i$  и  $B_i$  - антецедентные и последующие лингвистические термины (такие как «малые», «большие» и т. д.), представленные нечеткими множествами, а  $K$  - количество правил в модели. Лингвистическая нечеткая модель полезна для представления качественных знаний.

### Модель Такаги-Сугено (TS).

Модель Мамдани обычно используется в системах знаний (экспертных). В идентификации основанных на данных, модель Takagi и Sugeno стала популярной. В этой модели antecedent определяется так же, как и выше, в то время как последующая является линейной функцией входных переменных:

$$\mathcal{R}_i: \text{If } \mathbf{x} \text{ is } A_i \text{ then } y_i = \mathbf{a}_i^T \mathbf{x} + b_i, \quad i = 1, 2, \dots, K, \quad (2.3)$$

где  $a_i$  - последовательный параметр вектора, а  $b_i$  - скалярное смещение. Эта модель сочетает лингвистическое описание со стандартной функциональной регрессией: antecedенты описывают нечеткие области во входном пространстве, в котором действуют соответствующие функции. Результат  $y$  вычисляется путем вычисления взвешенного среднего вклада индивидуальных правил:

$$y = \frac{\sum_{i=1}^K \beta_i(\mathbf{x}) y_i}{\sum_{i=1}^K \beta_i(\mathbf{x})} = \frac{\sum_{i=1}^K \beta_i(\mathbf{x}) (\mathbf{a}_i^T \mathbf{x} + b_i)}{\sum_{i=1}^K \beta_i(\mathbf{x})} \quad (2.4)$$

где  $\beta_i(\mathbf{x})$  - степень выполнения  $i$ -го правила. Для правила указанного сверху,  $\beta_i(\mathbf{x}) = \mu_{A_i}(\mathbf{x})$ , но оно также может быть более сложным выражением, как показано ниже. Antecedentные нечеткие множества обычно определяются для описания отдельных частично перекрывающихся областей во входном пространстве. Параметры  $a_i$  тогда (приближенные) локальные линейные модели рассматриваемой нелинейной системы.

Таким образом, модель TS можно рассматривать как гладкую кусочно-линейную аппроксимацию нелинейной функции или модели планирования параметров. Обратим внимание, что antecedent и последующие переменные могут быть разными.

## Операторы нечеткой логики (AND, OR, NOT).

В нечетких системах с несколькими входами antecedентное предложение обычно представляется как комбинация терминов с одномерными функциями принадлежности, используя логические операторы “И” (конъюнкция), “ИЛИ” (дизъюнкция) и “НЕ” (дополнение). В теории нечетких множеств для этих логических связей введено несколько семейств операторов. В таблице 2.1 показаны наиболее распространенные.

$A \text{ and } B$	$A \text{ or } B$	$\text{not } A$
$\min(\mu_A, \mu_B)$	$\max(\mu_A, \mu_B)$	$1 - \mu_A$
$\mu_A \cdot \mu_B$	$\mu_A + \mu_B - \mu_A \cdot \mu_B$	$1 - \mu_A$

Таблица 2.1. Обычно используемые функции для операторов нечеткой логики.

## 2.2 Механизм нечеткого вывода

Система нечеткого вывода (FIS) по существу определяет нелинейное отображение вектора входных данных в скалярный вывод с использованием нечетких правил. Процесс сопоставления включает в себя функции ввода вывода, операторы FL, нечеткие правила if-then, агрегацию выходных множеств и дефаззификацию. FIS с несколькими выходами можно рассматривать как набор независимых многоканальных систем с одним выходом.



Рисунок 2.4. Общая модель системы нечеткого вывода (FIS).

FLS отображает четкие входы в четкие выходы. Из рисунка видно, что FIS содержит четыре компонента: *fuzzifier*, движок вывода, базу правил и *defuzzifier*. База правил содержит лингвистические правила, которые предоставляются экспертами. Также можно извлекать правила из числовых данных. Как только правила будут установлены, FIS можно рассматривать как систему, которая отображает входной вектор в выходной вектор. *Fuzzifier* отображает входные числа в соответствующие нечеткие членства. Это необходимо для того, чтобы активировать правила, относящиеся к лингвистическим переменным. *Fuzzifier* принимает входные значения и определяет степень, в которой они принадлежат каждому из нечетких множеств через функции принадлежности.

Механизм вывода определяет отображение входных нечетких множеств в выходные нечеткие множества. Он определяет степень, в которой антецедент выполняется для каждого правила. Если антецедент данного правила имеет более одного предложения, применяются нечеткие операторы, чтобы получить одно число, которое представляет результат антецедента для этого правила. Возможно, одно или несколько правил могут срабатывать одновременно. Затем выходы для всех правил агрегируются.

Во время агрегации нечеткие множества, представляющие вывод каждого правила, объединяются в один нечеткий набор. Нечеткие правила запускаются параллельно, что является одним из важных аспектов FIS. В FIS порядок, в котором запускаются правила, не влияет на выход. *Defuzzifier* отображает выходные нечеткие множества в четкое число.

Учитывая нечеткое множество, которое охватывает диапазон выходных значений, *defuzzifier* возвращает одно число, тем самым переходя от нечеткого множества к четкому числу.

На практике используются несколько методов дефазификации, в том числе центроид, максимум, среднее значение максимумов, высоты и измененного деформирующего устройства.

Самый популярный метод дефаззификации - это центроид, который вычисляет и возвращает центр тяжести агрегированного нечеткого множества. FIS используют правила. Однако, в отличие от правил в обычных экспертных системах, нечеткое правило локализует область пространства вдоль функциональной поверхности, а не изолирует точку на поверхности.

Для данного ввода может срабатывать более одного правила. Кроме того, в FIS множество областей объединяются в выходном пространстве для создания составной области.

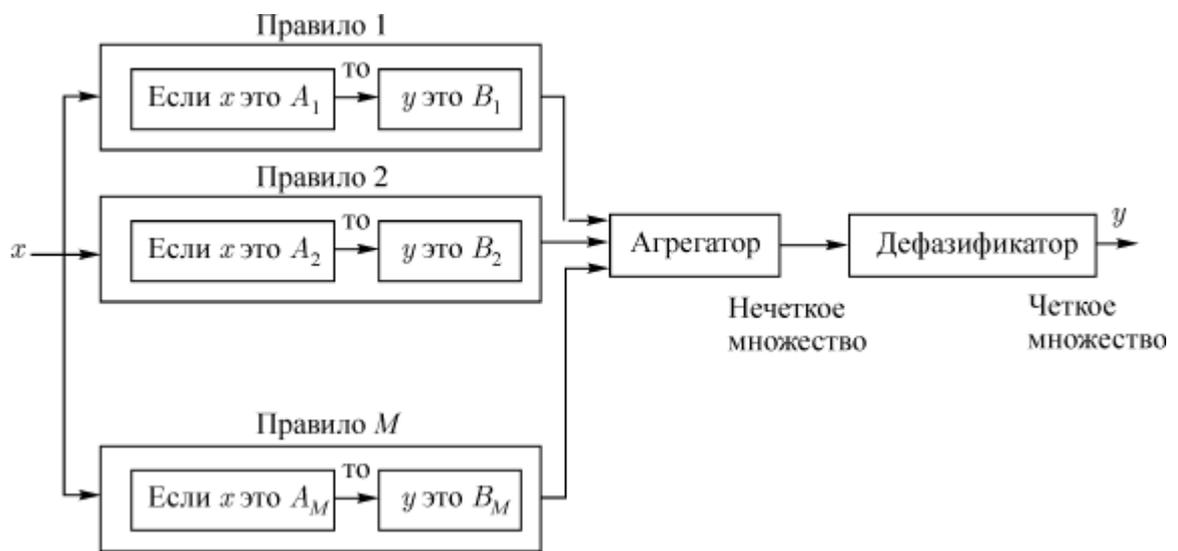


Рисунок 2.5. Общая схема FIS.

Рассмотрим многопроцессорную многоуровневую систему. Пусть  $\mathbf{x} = (x_1, x_2, \dots, x_n)^T$  – входной вектор, и  $\mathbf{y} = (y_1, y_2, \dots, y_m)^T$  – выходной вектор. Лингвистическая переменная  $x_i$  в пространстве  $U$  характеризуется

$$T(\mathbf{x}) = \{T_x^1, T_x^2, \dots, T_x^k\} \quad \text{и} \quad \mu(\mathbf{x}) = \{\mu_x^1, \mu_x^2, \dots, \mu_x^k\},$$

где  $T(\mathbf{x})$  – член множества  $x$ ; то есть это набор имен лингвистических значений  $x$ , причем каждый из них является нечетким членом  $T_x^i$  и функцией принадлежности  $\mu_x^i$ , определенной на  $U$ . В качестве иллюстрации мы

рассмотрим систему нечеткого вывода с двумя входами ( $n = 2$ ) и одним выходом ( $m = 1$ ). Пусть два входа представляют собой количество лет обучения и количество лет опыта, и пусть выпуск системы - это заработная плата. Пусть  $x_1$  указывает количество лет обучения,  $T(x_1)$  представляет собой его термин (низкий, средний, высокий) и пространство  $U [0-15]$ . Пусть  $x_2$  указывает количество лет опыта, пространство  $U [0-30]$ , а соответствующий термин – {низкий, средний, высокий}. Точно так же лингвистическая переменная  $y$  во вселенной дискурса  $V$  характеризуется  $T(y) = \{T_y^1, T_y^2, \dots, T_y^i\}$ , где  $T(y)$  – член множества  $y$ ; т. е.  $T$  - множество имен лингвистических значений  $y$ , с членом множества  $T_y^i$  и функцией принадлежности  $\mu_y^i$ , определенной на  $V$ .

Процесс нечеткого вывода.

Шаг 1. Нечеткие входы.

Первый шаг - принять входные данные и определить степень, в которой они принадлежат каждому из соответствующих нечетких множеств, через функции принадлежности.

Шаг 2. Применение нечетких операторов.

Как только входы были фаззификацированы, мы знаем степень, в которой каждая часть антецедента была удовлетворена для каждого правила. Если заданное правило имеет более одной части, нечеткие логические операторы применяются для оценки составной силы запуска правила.

Шаг 3: Применение метода импликации.

Метод импликации определяется как формирование выходных функций членства на основе силы возбуждения правила. Вход для процесса импликации - это одно число, заданное антецедентом, а выход - нечеткое множество. Двумя обычно используемыми методами импликации являются минимум и продукт.

#### Шаг 4: Агрегирование всех выходов.

Агрегация - это процесс, при котором результаты каждого правила унифицированы. Агрегация происходит только один раз для каждой выходной переменной. Ввод в процесс агрегации - это усеченные выходные нечеткие множества, возвращаемые процессом импликации для каждого правила. Результатом процесса агрегации является комбинированный выходной нечеткий набор.

#### Шаг 5: Defuzzify.

Вход для процесса дефаззификации - это нечеткое множество (агрегированный набор нечетких выходных данных), а выход процесса дефаззификации - это четкое значение, полученное с использованием некоторого метода дефаззификации, такого как центроид, высота или максимум.

Система нечеткого вывода отображает входной вектор в четкое выходное значение. Чтобы получить четкий выход, нам нужен процесс дефаззификации. Входом в процесс дефаззификации является нечеткое множество (агрегированный набор нечетких выходных данных), а выход процесса дефаззификации - это один номер. Предлагаются много методов дефаззификации. Наиболее часто используемый метод - это центроид. Другие методы включают в себя максимум, средство максимумов, высоты и метод измененной высоты. Пять методов могут быть описаны следующим образом:

Метод дефаззификации Centroid: в этом методе defuzzifier определяет центр тяжести  $Y_i'$  (центроид)  $B$  и использует это значение как результат FLS. Для непрерывного агрегированного нечеткого множества центроид задается формулой:

$$y' = \frac{\int_S y_i \mu_B(y) dy}{\int_S \mu_B(y) dy} \quad (2.5)$$

где, S обозначает носитель  $\mu_B(y)$ .

Часто используются дискретизированные переменные, так что  $y'$  можно аппроксимировать, как показано в уравнении ниже, в котором вместо интегрирования используются суммы.

$$y' = \frac{\sum_{i=1}^n y_i \mu_B(y_i)}{\sum_{i=1}^n \mu_B(y_i)} \quad (2.6)$$

Метод центроидного дефаззификации находит «точку равновесия» нечеткой области решения, вычисляя взвешенное среднее выходной нечеткой области. Это наиболее широко используемый метод, потому что, когда он используется, дефаззифицированные значения, как правило, плавно перемещаются вокруг выходной нечеткой области. Однако эта технология уникальна и нелегко реализовать вычислительно.

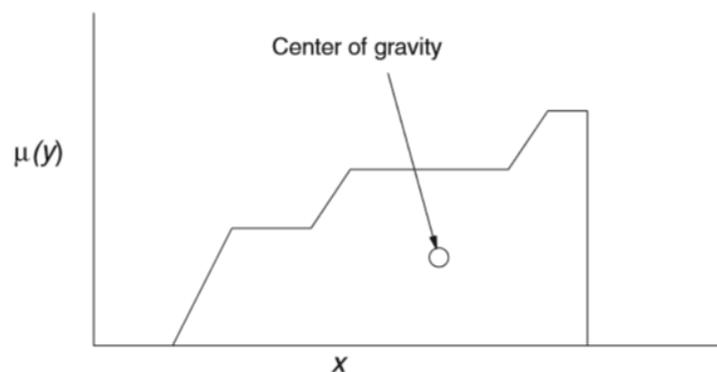


Рисунок 2.6. Метод центроидной дефаззификации.

Метод максимальной декомпозиции: в этом методе defuzzifier анализирует агрегированные нечеткие множества и выбирает такой выход  $y$ ,

для которого  $\mu_B(y)$  является максимальным. В отличие от метода центроидов, метод максимального разложения имеет некоторые свойства, применимые к более узкому классу проблем. Кроме того, выходное значение имеет тенденцию переходить от одного кадра к другому по мере изменения формы нечеткой области.

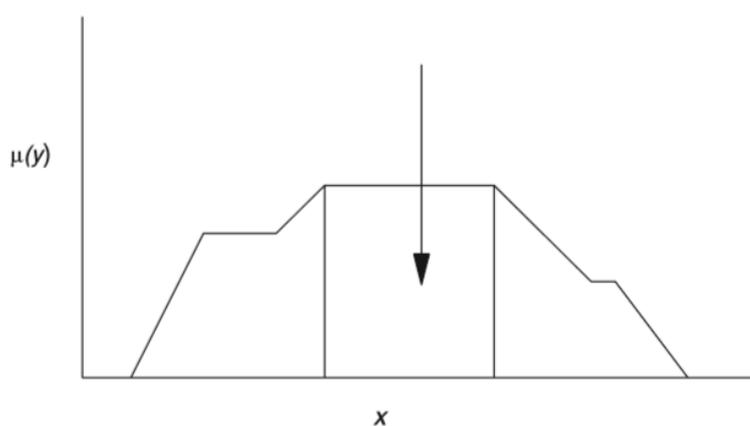


Рисунок 2.7. Метод максимальной декомпозиции.

Центр максимумов: в многорежимной нечеткой области техника центра максимумов находит самое высокое плато, а затем следующее самое высокое плато. И выбирается средняя точка между центрами этих плато.

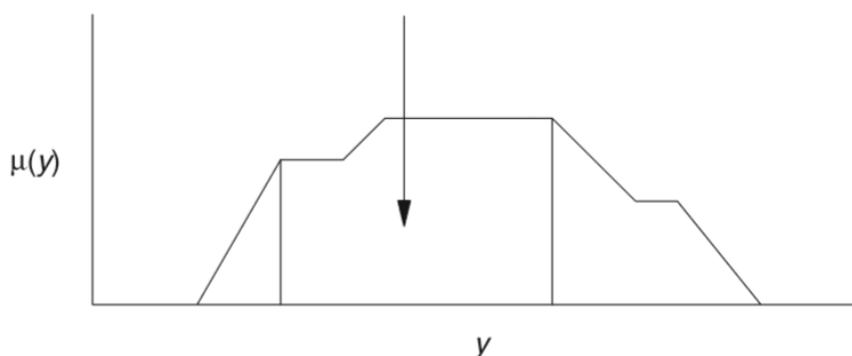


Рисунок 2.8. Метод центра максимумов.

Дефазификация по высоте. В этом методе дефазификатор сначала оценивает  $\mu_{B_i}(y)$  на  $y'_1$ , а затем вычисляет выход FLS, где определяется

центр тяжести нечетких множеств  $B_i$ . Выход  $y_h$  в этом случае задается выражением

$$y_h = \frac{\sum_{i=1}^m y_i' \mu_B(y_i)}{\sum_{i=1}^m \mu_B(y_i)} \quad (2.7)$$

где  $m$  представляет количество выходных нечетких множеств, полученных после импликации, а  $y_i'$  представляет центр тяжести нечеткой области  $i$ . Этот метод прост в использовании, поскольку центры тяжести широко используемых членов функций известны заранее.

Независимо от того, используется ли минимальный или производственный вывод, процесс нечеткого вывода существенно определяет отображение данного вектора хрустящих значений на выходное четкое значение с использованием нечетких правил, хранящихся в базе знаний.

Рассматриваемый процесс нечеткого вывода известен как метод нечеткого вывода Мамдани. Сугено (1977) предложил метод нечеткого вывода, аналогичный методу Мамдани.

### 2.3 Нейронные сети

Существуют категории проблем, которые нельзя сформулировать как алгоритм. Проблемы, связанные с уменьшением количества факторов, например, покупная цена недвижимости, которую наш мозг может (приблизительно) рассчитать. Без алгоритма компьютер не может сделать то же самое. Мозг содержит  $10^{11}$  нейронов, но они имеют время переключения около  $10^{-3}$  секунд. Большая часть мозга работает непрерывно, в то время как большая часть компьютера – это только пассивное хранилище данных. Мозг как биологическая нейронная сеть может реорганизоваться в течение своей «продолжительности жизни» и, следовательно, способен учиться, компенсировать ошибки и т. д.

Таким образом, изучение искусственных нейронных сетей мотивировано их сходством с успешной работой биологических систем, которые, по сравнению с общей системой, состоят из очень простых, но многочисленных нервных клеток, которые работают в широких пределах (что, вероятно, является одним из наиболее значимых аспектов), имеют возможность учиться. Нет необходимости явно программировать нейронную сеть.

Например, он может учиться на учебных образцах или посредством поощрения, так сказать (усиление обучения). Одним из результатов этой учебной процедуры является способность нейронных сетей обобщать и ассоциировать данные: после успешного обучения нейронная сеть может найти разумные решения для аналогичных проблем того же класса, которые не были явно обучены. Это, в свою очередь, приводит к высокой степени отказоустойчивости от шумных входных данных. Отказоустойчивость тесно связана с биологическими нейронными сетями, в которых эта характеристика очень различна: как известно, человек имеет около  $10^{11}$  нейронов, которые постоянно реорганизуются.

Поле нейронных сетей, как и любое другое направление науки, имеет долгую историю развития со многими взлетами и падениями. История нейронных сетей начинается в начале 1940-х годов и, следовательно, почти одновременно с историей программируемых электронных компьютеров, и является изучаемым и применяемым в современное время.

Переход от биологических нейронов к искусственным. В биологических нейронах есть привязанность друг к другу, и когда они стимулируются, они электрически передают свой сигнал через аксон. Из аксона они не переносятся непосредственно в последующие нейроны, но сначала им приходится пересекать синаптическую щель, где сигнал снова изменяется с помощью переменных химических процессов.

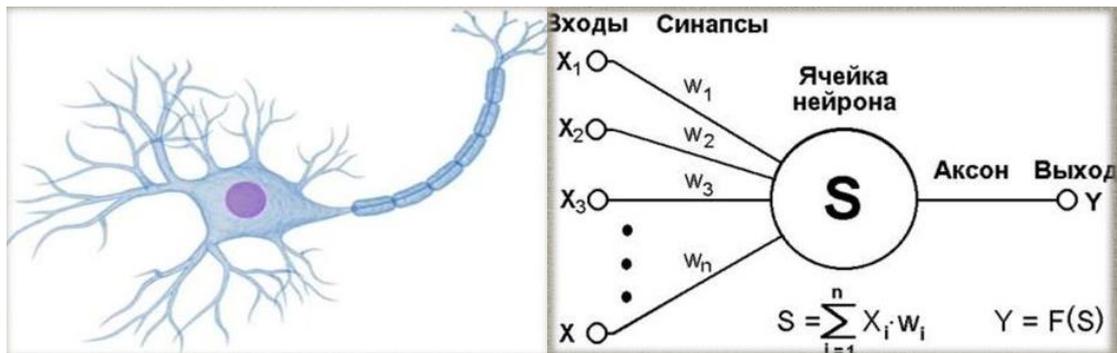


Рисунок 2.9. Биологическая нейронная сеть и искусственная нейронная сеть.

В принимающем нейроне различные входы, которые были обработаны постобработкой в синаптической щели, суммируются или накапливаются до одного импульса. Краткая сводка точно соответствует нескольким элементам биологических нейронных сетей, которые мы хотим взять в техническое приближение:

**Векторный ввод:** ввод технических нейронов состоит из многих компонентов, поэтому он является вектором. В природе нейрон получает импульсы от  $10^3$  до  $10^4$  других нейронов в среднем.

**Скалярный выход:** выход нейрона является скаляром, что означает, что нейрон состоит только из одного компонента. Несколько скалярных выходов в свою очередь образуют векторный вход другого нейрона. Это, в частности, означает, что где-то в нейроне различные входные компоненты могут быть объединены таким образом, что остается только один компонент.

**Ввод изменений в синапсах:** в технических нейронных сетях ввод также предварительно обрабатывается. Они умножаясь на число (вес) – взвешиваются. Набор таких весов представляет собой хранение информации нейронной сети – как в биологической оригинальной, так и в технической адаптации.

**Накопление входов:** в биологии входы суммируются с импульсом в соответствии с химическим изменением, т.е. они накапливаются, с технической стороны это часто реализуется взвешенной суммой, о которой мы

узнаем позже. Это означает, что после накопления мы получаем только одно значение – скаляр вместо вектора.

Нелинейная характеристика: ввод наших искусственных нейронов также не пропорционален выходу.

Регулируемые веса: веса, взвешивающие входы, являются переменными, аналогичным химическим процессам в синаптической щели. Это добавляет большую динамику в сеть, поскольку значительная часть «знаний» нейронной сети сохраняется в весах, а также в форме и силе химических процессов в синаптической щели.

Таким образом, наша текущая, только случайно сформулированная и очень простая нейронная модель получает векторный вход  $\vec{x}$  с компонентами  $x_i$ . Они умножаются на соответствующие весовые коэффициенты  $w_i$  и накапливаются:

$$\sum_i w_i x_i. \quad (2.8)$$

Вышеупомянутый термин называется взвешенной суммой. Тогда нелинейное отображение  $f$  определяет скалярный выход  $y$ :

$$y = f \left( \sum_i w_i x_i \right) \quad (2.9)$$

Концепция времени в нейронных сетях.

В некоторых определениях этого текста мы используем термин время или число циклов нейронной сети, соответственно. Время делится на отдельные временные шаги:

- текущее время (в настоящее время) определяется как  $(t)$ ,
- следующий шаг времени как  $(t + 1)$ ,
- предыдущий шаг как  $(t - 1)$ .

Все остальные временные шаги упоминаются аналогично.

Искусственная нейронная сеть состоит из простых блоков обработки, нейронов и направленных взвешенных связей между этими нейронами. Здесь сила соединения (или связующего веса) между двумя нейронами  $i$  и  $j$  определяется как  $w_{i,j}$ .

Нейронная сеть представляет собой отсортированную тройку  $(N, V, w)$  с двумя множествами  $N, V$  и функцией  $w$ , где  $N$  - множество нейронов и  $V$  множество  $\{(i, j) \mid i, j \in N\}$ , элементы называются связями между нейроном  $i$  и нейроном  $j$ .

Функция  $w : V \rightarrow \mathbb{R}$  определяет веса, где  $w((i, j))$ , вес связи между нейроном  $i$  и нейроном  $j$  сокращается до  $w_{i,j}$ . Для соединений которых не существует в сети, он принимает значение 0. Соединения содержат информацию, обрабатываемую нейронами. Данные переносятся между нейронами через веса связи.

Нейронные сети состоят из слоев и соединений по отношению к каждому следующему слою. Нейроны сгруппированы в следующие слои:

- один входной слой;
- $n$  скрытых слоев обработки (невидимый снаружи, поэтому нейроны также называются скрытыми нейронами);
- один выходной слой.

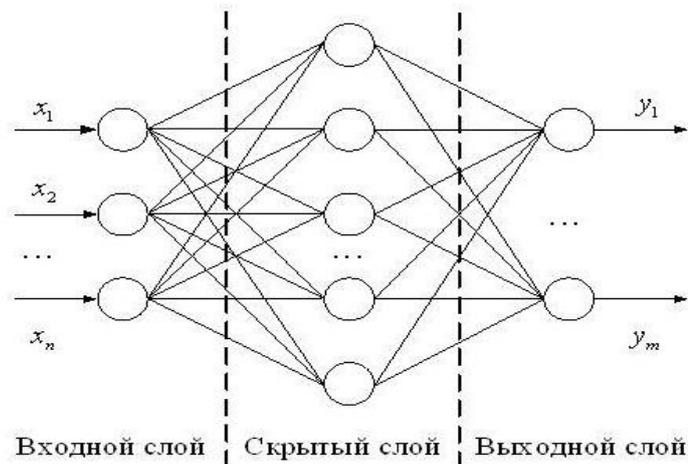


Рисунок 2.10. Слои нейронных сетей.

## Топология нейронных сетей.

Существуют топологии нейронных сетей:

- Feedforward (с прямой связью);
- Feedback (с обратной связью);
- Сеть “Hopfield” (с симметрической связью).

В прямой сети каждый нейрон в одном слое имеет только направленные соединения с нейронами следующего слоя (к выходному слою). Мы часто сталкиваемся с прямыми сетями, в которых каждый нейрон связан со всеми нейронами следующего слоя (эти слои называются полностью связанными).

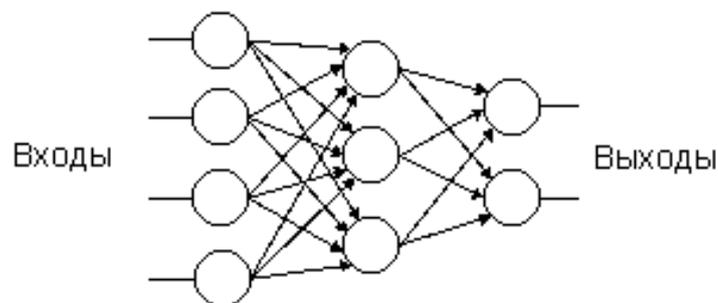


Рисунок 2.11. Нейронная сеть “feedforward”.

Уровни нейронов сети *feedforward* четко разделены: один входной слой, один выходной слой и один или несколько слоев обработки, которые невидимы снаружи (также называемые скрытыми слоями). Соединения разрешены только для нейронов следующего слоя. В сети с обратной связью, выходной нейрон передает результат другому нейрону находящемуся на том или на предыдущем уровне.

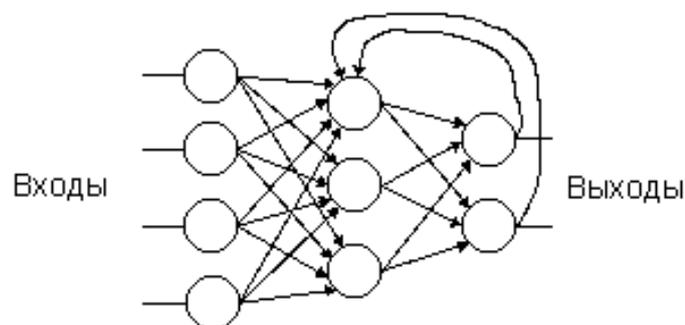


Рисунок 2.12. Нейронная сеть “feedback”.

Нейронная сеть “*Hopfield*” обладает одним слоем, где каждый нейрон может быть как входом, так и выходом. Говоря кратко, сеть “*Hopfield*” состоит из множества  $K$  полностью связанных нейронов с бинарной активацией, причем веса симметричны между отдельными нейронами и вне зависимости от того, какой нейрон связан с собой. Функция активации нейронов является бинарной пороговой функцией с выходами  $\in \{1, -1\}$ . Таким образом,  $|K|$  нейронов с двумя возможными состояниями  $\in \{-1, 1\}$  могут быть описаны  $x \in \{-1, 1\}^{|K|}$ .

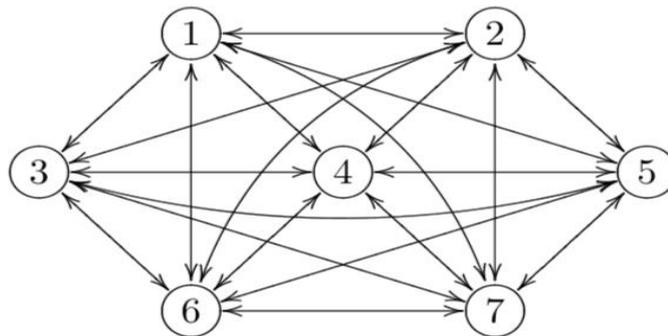


Рисунок 2.13. Сеть “*Hopfield*” или сеть с симметрической связью.

## 2.4 Нейросетевое моделирование информационных процессов

Нейронные сети становятся все более популярными для задачи моделирования информационных процессов. Рассмотрим модели нейронных сетей, которые используются как в отдельности, так и комбинированно для моделирования.

### Модель сети “Хемминга”

Сети Хэмминга были спроектированы явно для решения проблемы распознавания двоичных образов (где каждый элемент входного вектора, имеет только два возможных значения, такие как 1 или -1).

Эта сеть называется сетью Хэмминга, потому что нейрон в слое с прямой связью с наибольшим выходом будет соответствовать шаблону прототипа, который наиболее близок в расстоянии Хэмминга к шаблону ввода. (Расстояние Хэмминга между двумя векторами равно числу элементов,

которые отличаются друг от друга. Это определенные только для двоичных векторов.) Нейроны в слое feedback инициализируются выходами слоя feedforward, которые указывает на корреляцию между шаблонами прототипов и входной вектор. Затем нейроны конкурируют друг с другом, чтобы определить победителя. После этого только один нейрон имеет выход отличный от нуля. Победивший нейрон указывает, какая категория ввода была представлена в сеть. Корреляция - это единственное число, которое описывает степень сходства между двумя переменными / образцами:

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}} \quad (2.10)$$

Сеть включает как feedforward, так и feedback слои.

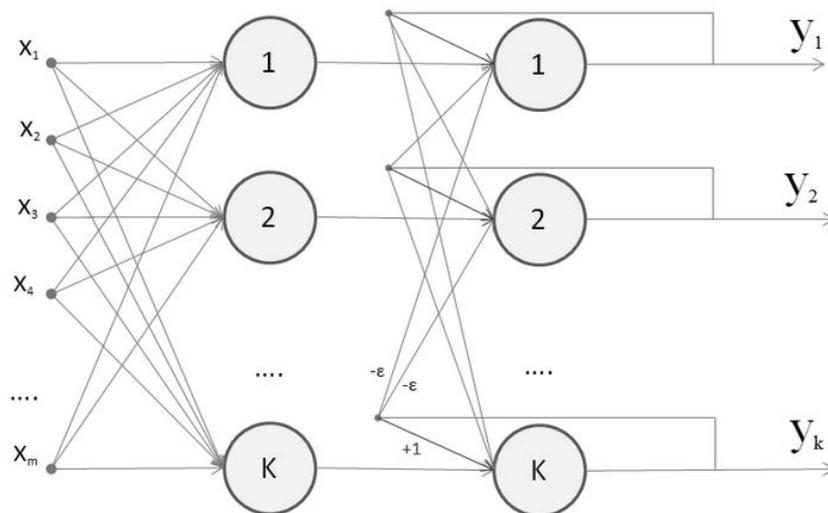


Рисунок 2.14. Сеть “Хемминга”.

### Модель сети “Кохонена”.

Самой известной и самой популярной моделью самоорганизующихся сетей является карта сохранения топологии, предложенная Теуво Кохоненом. Так называемые сети Кохонена являются воплощением некоторых идей,

разработанных Розенблаттом, Фон-дер-Малсбургом и другими исследователями. Если входное пространство должно обрабатываться нейронной сетью, первым вопросом важности является структура этого пространства.

Нейронная сеть с реалистичными вычислениями вычисляет функцию  $f$ , определяемую из входного пространства  $A$ , в выходное пространство  $B$ . Область, где  $f$  определена, может быть покрыта сетью Кохонена. Сети Кохонена учатся создавать карты входного пространства самоорганизующимся способом.

Модель Кохонена имеет биологический и математический фон. В нейробиологии хорошо известно, что многие структуры мозга имеют линейную или плоскую топологию, то есть они распространяются в одном или двух измерениях. Сети Кохонена представляют собой устройства вычислительных узлов в одно-, двух- или многомерных решетках. У узлов есть боковые связи с несколькими соседями.

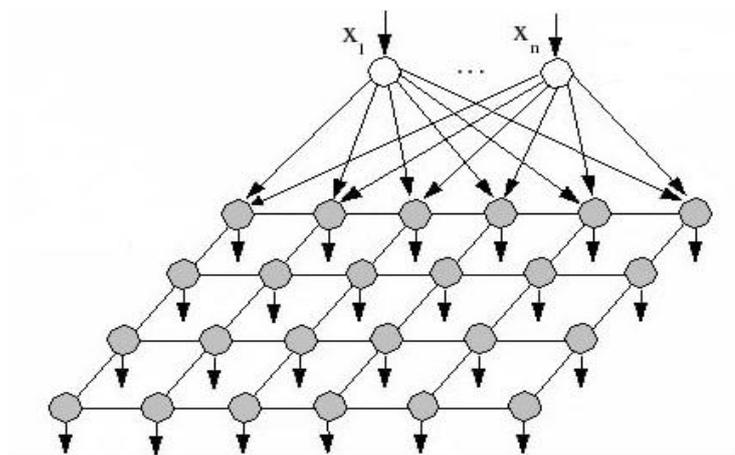


Рисунок 2.15. Сеть “Кохонена”

Примерами такого рода боковых связей являются ингибирующие связи. Входной нейрон соединяется с первым узлом (нейроном) слоя Кохонена, весом  $w_{A1-B1}$ . Каждый узел выдает – один выходной нейрон.

$$OUT_k = w_{A1-Bk} \times x_1 + w_{A2-Bk} \times x_2 + w_{A3-Bk} \times x_3 \quad (2.11)$$

или

$$\sum_n^{n=1..3} w_{An-Bk} \times x_n \quad (2.12)$$

где  $OUT_k$  является выходным нейроном слоя Кохонена.

Сетка вычислительных элементов позволяет идентифицировать непосредственных соседей. Это очень важно, поскольку во время обучения весовые вычислительные единицы и их соседи были обновлены. Объектом такого подхода к обучению является то, что соседние подразделения учатся реагировать на тесно связанные сигналы.

### **Перцептрон: однослойный и многослойный.**

Как уже упоминалось в истории нейронных сетей, перцептрон был описан Фрэнком Розенблаттом в 1958 году. Первоначально Розенблатт определял уже обсуждаемую взвешенную сумму и нелинейную активационную функцию, как компоненты перцептрона.

Не существует установленного определения для перцептрона, но большую часть времени этот термин используется для описания сети с прямым соединением с помощью ярлыков. Эта сеть имеет слой нейронов сканера (сетчатка) со статически взвешенными соединениями со следующим слоем, называемым входным слоем; но вес всех остальных слоев может быть изменен. Все нейроны, подчиненные сетчатке, являются детекторами.

Здесь мы первоначально используем бинарный перцептрон с каждым выходным нейроном, имеющим ровно два возможных значения (например,  $\{0,1\}$  или  $\{-1,1\}$ ).

Таким образом, бинарная функция порога используется как функция активации в зависимости от порогового значения выходного нейрона. В некотором смысле функция двоичной активации представляет собой запрос

$IF$ , который также можно отрицать с помощью отрицательных весов. Таким образом, перцептрон можно использовать для обработки истинной логической информации.

Перцептрон представляет собой прямую сеть, содержащую сетчатку (сканер), которая используется только для сбора данных и которая имеет фиксированные взвешенные соединения с первым нейронным слоем (входным слоем). За фиксированным весовым слоем следует, по меньшей мере, один обучаемый весовой слой. Один слой нейрона полностью связан со следующим слоем.

Первый слой перцептрона состоит из входных нейронов. Здесь соединения с обучаемыми весами идут от входного уровня к выходному нейрону, который возвращает информацию о том, был ли признан шаблон, введенный во входные нейроны, или нет.

Таким образом, однослойное восприятие (сокращенно SLP) имеет только один уровень тренируемых весов. Однослойный перцептрон (SLP) является перцептроном, имеющим только один слой переменных весов и один слой выходных нейронов.

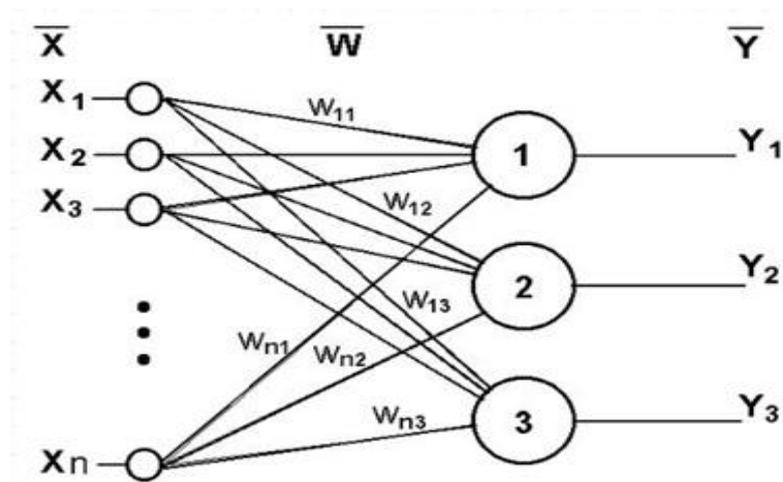


Рисунок 2.16. Однослойный перцептрон.

Перцептрон с двумя или более тренируемыми весовыми слоями является более мощным, чем SLP. Перцептроны с более чем одним слоем переменных взвешенных соединений называются многослойными перцептронами (MLP).

Таким образом,  $n$ -слойный или  $n$ -ступенчатый персептрон имеет ровно  $n$  слоев с переменным весом и  $n + 1$  нейронных слоев (здесь не учитывается сетчатка), причем слой нейрона  $1$  является входным слоем.

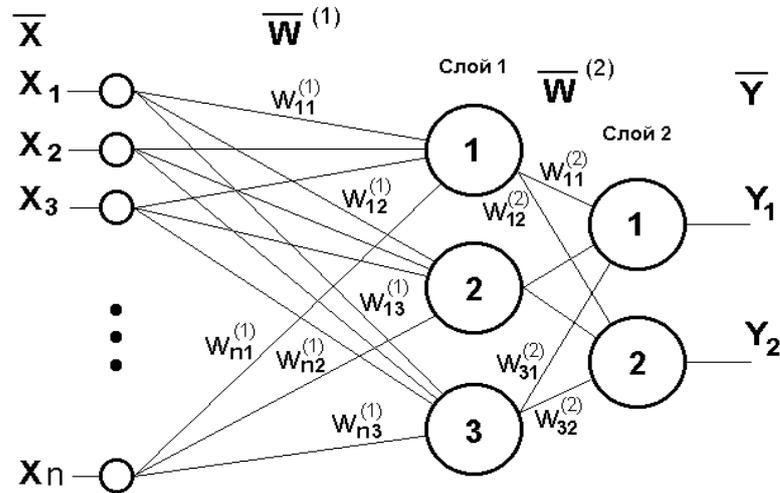


Рисунок 2.17. Многослойный персептрон.

Из этого следует что  $n$ -этапный персептрон имеет ровно  $n$  обучаемых весовых слоев.

Все вышеперечисленные модели нейронной сети применяются в системах для моделирования информационных процессов, независимо от применения этих моделей как в отдельности, так и слиянием, т.е. гибридами этих моделей.

На основе этих вышеперечисленных моделей были созданы такие гибридные модели, которые также можно реализовать в среде MATLAB, с высокой точностью и реализацией процессов.

Нейронные сети применяются для моделирования таких слабоструктурированных процессов, как распознавание голосов и звуков, прогнозирование финансовых временных рядов и т.д. Основой является выбор самой простой модели для решения поставленной задачи.

## ГЛАВА III. НЕЙРО-НЕЧЕТКИЕ МЕТОДЫ И СРЕДСТВА МОДЕЛИРОВАНИЯ СЛАБОСТРУКТУРИРОВАННЫХ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ

### 3.1. Нейро-нечеткое моделирование

Soft-computing включают в себя нечеткую логику, нейронные сети, вероятностные рассуждения и генетические алгоритмы. Сегодня методы или комбинация методов из всех этих областей используются для разработки разведывательной системы. Нейронные сети обеспечивают алгоритмы обучения, классификации и оптимизации, тогда как нечеткая логика имеет дело с такими вопросами, как формирование впечатлений и рассуждений на семантическом или лингвистическом уровне.

Вероятностные рассуждения касаются неопределенности. Хотя существуют существенные области перекрытия между нейронными сетями, FL и вероятностными рассуждениями, в целом они скорее дополняют друг друга, чем конкурируют. В последнее время использовались многие интеллектуальные системы, называемые нейро-нечеткими системами. Существует множество способов комбинирования нейронных сетей и методов FL.

Нейро-нечеткое моделирование было признано мощным инструментом, который может способствовать эффективной разработке моделей путем объединения информации из разных источников, таких как эмпирические модели, эвристика и данные.

Нейро-нечеткие модели описывают системы с помощью нечетких правил if-then, таких как «Если  $x$  маленький, то  $y$  является большим», представленным в сетевой структуре, к которой могут быть применены алгоритмы обучения, известные из области ИНС. Благодаря этой структуре нейро-нечеткие модели в определенной степени прозрачны для интерпретации

и анализа, т. е. могут быть лучше применены для объяснения решений для пользователей, чем "черный ящик", с применением нейронных сетей.

И нейронные сети, и нечеткие системы мотивированы имитацией процессов человеческого мышления. В нечетких системах отношения явно представлены в виде правил if-then. В нейронных сетях отношения не указаны явно, а «закодированы» в сети и ее параметрах.

В отличие от методов, основанных на знаниях, явное знание не требуется для применения нейронных сетей. Нейро-нечеткие системы объединяют семантическую прозрачность нечетких систем на основе правил с возможностями обучения нейронных сетей.

На вычислительном уровне нечеткую систему можно рассматривать как многоуровневую структуру (сеть), аналогичную искусственным нейронным сетям типа RBF. Для оптимизации параметров в нечеткой системе могут использоваться градиентные методы обучения, известные из области нейронных сетей.

Следовательно, этот подход обычно упоминается как нейро-нечеткое моделирование. Рассмотрим сначала простой пример нечеткой модели TS нулевого порядка со следующими двумя правилами:

$$\begin{aligned} \text{If } x_1 \text{ is } A_{11} \text{ and } x_2 \text{ is } A_{21} \text{ then } y &= b_1 \\ \text{If } x_1 \text{ is } A_{12} \text{ and } x_2 \text{ is } A_{22} \text{ then } y &= b_2 \end{aligned} \quad (3.1)$$

На рисунке 3.1 показано сетевое представление этих двух правил. Узлы в первом слое вычисляют степень принадлежности входов в antecedentes нечетких множествах. Узлы продукта  $\Pi$  во втором слое представляют собой antecedентную связку (здесь оператор «и»). Узел нормировки  $N$  и узел суммирования  $\Sigma$  реализуют нечеткий средний оператор.

Эта система называется ANFIS - Adaptive NeuroFuzzy Inference System.

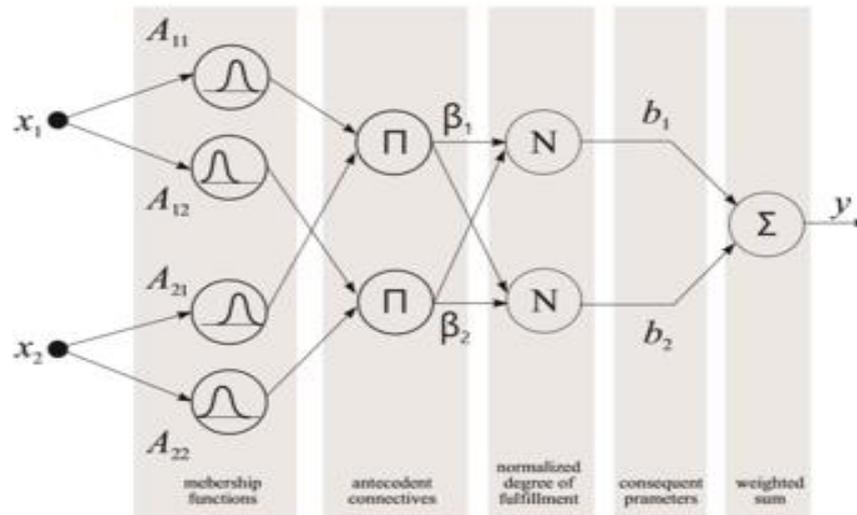


Рисунок 3.1. Пример нечеткой модели TS нулевого порядка с двумя правилами, представленными в виде нейро-нечеткой сети.

Как правило, используются гладкие antecedentesные функции принадлежности, такие как гауссовские функции:

$$\mu_{A_{ij}}(x_j; c_{ij}, \sigma_{ij}) = \exp\left(-\frac{(x_j - c_{ij})^2}{2\sigma_{ij}^2}\right) \quad (3.2)$$

Уравнение вход-выход общей модели TS нулевого порядка с конъюнктивной формой antecedента:

$$y = \sum_{i=1}^K \gamma_i(\mathbf{x}) b_i \quad \text{with} \quad \gamma_i(\mathbf{x}) = \frac{\prod_{j=1}^p \exp\left(-\frac{(x_j - c_{ij})^2}{2\sigma_{ij}^2}\right)}{\sum_{i=1}^K \prod_{j=1}^p \exp\left(-\frac{(x_j - c_{ij})^2}{2\sigma_{ij}^2}\right)} \quad (3.3)$$

Аналогичную модель можно представить в виде нечеткой модели TS первого порядка. Рассмотрим снова пример с двумя правилами:

$$\begin{aligned} \text{If } x_1 \text{ is } A_{11} \text{ and } x_2 \text{ is } A_{21} \text{ then } y_1 &= a_{11}x_1 + a_{12}x_2 + b_1 \\ \text{If } x_1 \text{ is } A_{12} \text{ and } x_2 \text{ is } A_{22} \text{ then } y_2 &= a_{21}x_1 + a_{22}x_2 + b_2 \end{aligned} \quad (3.4)$$

для которых соответствующая сеть приведена на рисунке 3.2.

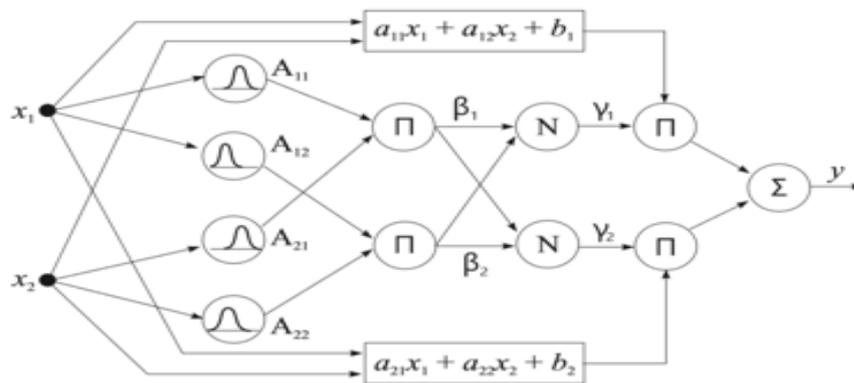


Рисунок 3.2. Пример нечеткой модели TS первого порядка с двумя правилами, представленными в виде нейро-нечеткой сети, называемой ANFIS.

Уравнение вход-выход модели TS первого порядка:

$$y = \sum_{i=1}^K \gamma_i(\mathbf{x}) (\mathbf{a}_i^T \mathbf{x} + b_i) \quad (3.5)$$

Далее рассмотрим нейро-нечеткие модели с их архитектурой.

### **Обобщенно-ориентировочное подход на основе интеллектуального управления (модель GARIC).**

GARIC – это расширенная версия ориентировочный подход Verenjeji на основе интеллектуального управления, который реализует нечеткий контроллер, используя несколько специализированных нервных сетей с прямой передачей. Он состоит из сети оценки состояния действия (AEN) и сети выбора действий (ASN). Архитектура GARICASN изображена на рисунке 3.3.

ASN GARIC - это сеть с прямой связью с пятью слоями.

1-ый скрытый слой хранит лингвистические значения всех входных переменных. Каждый блок ввода подключен только к тем единицам 1-го скрытого слоя, которые представляют собой связанные с ним лингвистические значения.

2-ой скрытый слой представляет узлы нечетких правил, которые определяют степень выполнения правила с помощью операции  $\text{softmin}$ .

3-ий скрытый слой представляет лингвистические значения управляющей выходной переменной  $\eta$ .

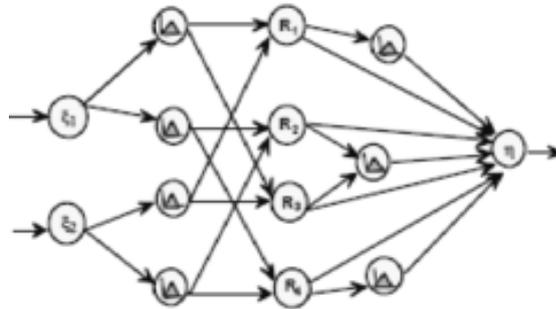


Рисунок 3.3. Архитектура GARICASN.

Выводы правила вычисляются в зависимости от силы antecedентов правил, вычисленных на уровне узла правила. GARIC использует метод локального среднего максимума для вычисления выходов правил. Этот метод требует четкого выходного значения из каждого правила.

Поэтому выводы должны быть деактивированы до того, как они будут накоплены до конечного выходного значения контроллера. GARIC использует смесь градиентного спуска и обучения для усиления точной настройки параметров узла. Гибридное обучение прекращается, если выход AEN перестает меняться. Очень трудоемкий процесс обучения и сложную архитектуру GARIC можно считать главными недостатками GARIC.

### **Нейро-нечеткое управление (модель NEFCON).**

Алгоритм обучения, определенный для NEFCON, способен изучать нечеткие множества, а также нечеткие правила, реализующие FIS типа Mamdani. Этот метод можно рассматривать как расширение для GARIC, которое также использует обучение подкрепления, но нуждается в ранее определенной базе правил. На рисунке 3.4 показана базовая архитектура NEFCON с двумя входами и пятью нечеткими правилами.

Внутренние узлы  $R_1, \dots, R_5$  представляют собой правила, узлы  $\xi_1, \xi_2$  и  $\eta$  - входные и выходные значения, а  $\mu_r, V_r$  - нечеткие множества, описывающие antecedentes и следствия. В отличие от нейронных сетей, соединения в NEFCON взвешиваются с нечеткими множествами вместо реальных чисел. Правила с тем же antecedентом используют так называемые общие веса, которые представлены эллипсами, нарисованными вокруг соединений, как показано на рисунке 3.4. Они обеспечивают целостность базы правил. База знаний нечеткой системы неявно определяется структурой сети.

Модули ввода принимают на себя задачу интерфейса фазификации, логика вывода представлена функциями распространения, а выходным модулем является интерфейс дефазификации.

Процесс обучения модели NEFCON можно разделить на две основные фазы. Обучение инкрементному правилу используется, когда корректный вывод неизвестен и правила создаются на основе оценочных выходных значений. По мере прохождения обучения, в соответствии с требованием добавляется больше правил.

Для обучения декрементному правилу изначально правила создаются из нечетких разделов переменных процессов, а ненужные правила устраняются в процессе обучения. Обучение декрементному правилу менее эффективно по сравнению с инкрементным подходом.

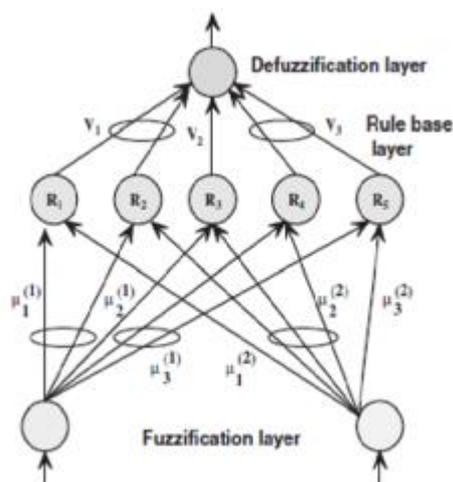


Рисунок 3.4. Архитектура NEFCON.

Из-за сложности требуемых вычислений правило декрементного обучения может быть использовано только в том случае, если имеется только несколько входных переменных с не слишком большим количеством нечетких множеств. Для более крупных систем правило инкрементного обучения будет оптимальным.

Предварительное знание, когда оно доступно, может быть включено для снижения сложности обучения. Функции принадлежности в базе правил изменяются в соответствии с алгоритмом распространения нечеткой ошибки (FEBP). Алгоритм FEBP может адаптировать функции принадлежности и может применяться только в том случае, если уже существует база нечетких правил.

Идея алгоритма обучения идентична: увеличивает влияние правила, если его действие идет в правильном направлении (вознаграждение) и уменьшает его влияние, если правило ведет себя продуктивно (наказывает). Если нет абсолютно никакой информации о начальной функции принадлежности, следует использовать равномерное нечеткое разбиение переменных.

### **Развивающиеся (Эволюционные) нечеткие нейронные сети (модели EFuNN и dmEFuNN).**

EFuNN и dmEFuNN основаны на структуре ECOS (эволюционирующих систем соединений) для адаптивных интеллектуальных систем, сформированных из-за эволюции и инкрементного, гибридного (контролируемого/неконтролируемого) онлайн обучения. Они могут размещать новые входные данные, включая новые функции, новые классы и т.д. посредством локальной настройки элементов. В EFuNN все узлы создаются во время обучения. EFuNN имеет пятислойную архитектуру, как показано на рисунке 3.5. Входной уровень представляет собой буферный слой, представляющий входные переменные. Второй уровень узлов представляет собой нечеткое квантование каждого входного пространства переменных. Каждая входная переменная представлена здесь группой пространственно-

расположенных нейронов для представления нечеткого квантования этой переменной. Узлы, представляющие функции принадлежности (треугольные, гауссовские и т. д.), могут быть изменены во время обучения.

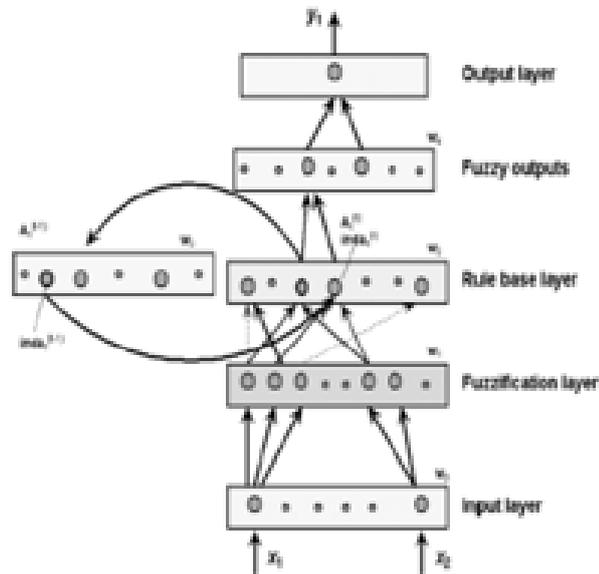


Рисунок 3.5. Архитектура EFuNN.

Третий уровень содержит узлы правил, которые развиваются через гибридное (контролируемое /неконтролируемое) обучение.

Узлы правил представляют собой прототипы ассоциаций данных ввода-вывода, графически представляемых как ассоциация гиперсфер из нечетких входных и нечетких выходных пространств. Каждый узел правила  $r$  определяется двумя векторами весов соединения:  $W_1(r)$  и  $W_2(r)$ , причем последний корректируется посредством контролируемого обучения на основе ошибки выхода, а первая корректируется с помощью неконтролируемого обучения на основе измерения сходства в локальной области пространства проблемы ввода.

Четвертый слой нейронов представляет собой нечеткое квантование для выходных переменных.

Пятый уровень представляет собой реальные значения для выходных переменных. В случае «oneof-n» EFuNN максимальная активация узла

правила распространяется на следующий уровень. В случае режима «many-of-n» все значения активации узлов правил, которые превышают порог активации, далее распространяются в структуре соединения.

Динамическая эволюционная модель нечетких нейронных сетей (dmEFuNN) разработана с мыслью, что распространяется не только активация узла выигрышного правила, но и группа узлов правил динамически выбранных для каждого нового входного вектора, и их значения активации используются для расчета динамических параметров выходной функции. В то время как EFuNN использует взвешенные нечеткие правила типа Mamdani, dmEFuNN использует нечеткие правила Takagi-Sugeno. Архитектура изображена на рисунке 3.6.

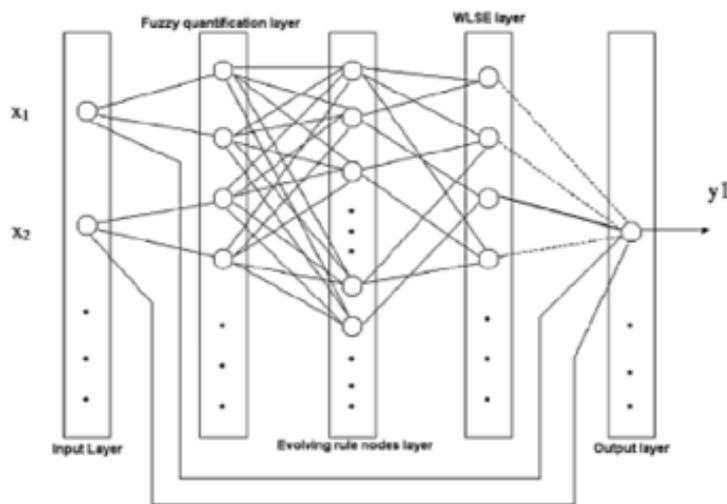


Рисунок 3.6. Архитектура dmEFuNN.

Первый, второй и третий уровни dmEFuNN имеют точно такие же структуры и функции, что и EFuNN.

Четвертый слой, слой нечеткого вывода, выбирает  $m$  узлов правил из третьего уровня, которые имеют самое близкое нечеткое нормированное локальное расстояние до нечеткого входного вектора, а затем, нечеткое правило Takagi Sugeno сформируется с использованием взвешенной оценки наименьших квадратов.

Последний уровень вычисляет вывод  $dmEFuNN$ . Число  $m$  активированных узлов, используемых для вычисления выходных значений  $dmEFuNN$ , не меньше числа входных узлов плюс один. Как и  $EFuNN$ ,  $dmEFuNN$  могут использоваться как для автономного обучения, так и для онлайн-обучения, таким образом оптимизируя глобальную ошибку обобщения или локальную ошибку обобщения.

В  $dmEFuNN$  для нового входного вектора (для которого выходной вектор неизвестен), найдено подпространство, состоящее из  $m$  узлов правила, формируется нечеткое правило Takagi Sugeno первого порядка с использованием метода наименьших квадратов. Это правило используется для вычисления выходного значения  $dmEFuNN$ .

Таким образом,  $dmEFuNN$  действует как универсальный аппроксиматор функции, используя  $m$  линейных функций в малом  $m$ -мерном узле. Точность приближения зависит от размера узловых подпространств, чем меньше подпространство, тем выше точность.

### **3.2. Моделирование слабоструктурированных процессов методами и средствами ANFIS**

#### **Построение Нейро-нечеткой сети.**

Как предварительные знания, так и данные процесса могут быть использованы для построения нейро-нечетких систем. Предварительное знание может иметь довольно приблизительный характер (качественный, эвристический). Можно выделить два основных подхода к интеграции знаний и данных:

1. Знание эксперта формулируется как совокупность правил *if-then*. Таким образом создается исходная модель. Параметры этой модели (функции принадлежности, последующие параметры) затем настраиваются с использованием данных процесса.

2. Нечеткие правила (включая связанные параметры) строятся с нуля с использованием числовых данных. В этом случае преимуществом использования нейро-нечеткой модели является возможность интерпретировать полученный результат (что невозможно с “черными ящиками”, как нейронные сети).

Эксперт может справиться с информацией, хранящейся в базе правил, своими собственными знаниями, изменять правила или предоставлять дополнительные, чтобы продлить срок действия модели и т. д.

Вышеупомянутые методы, конечно, могут быть объединены в зависимости от проблемы.

### **Структура и параметры.**

Двумя основными этапами системной идентификации являются структурная идентификация и оценка параметров. Выбор структуры модели очень важен, поскольку он определяет гибкость модели в приближении (неизвестных) систем. Модель с богатой структурой может аппроксимировать более сложные функции, но в то же время будет иметь худшие свойства обобщения. Хорошее обобщение означает, что модель, привязанная к одному множеству данных, также хорошо работает на другом множестве данных в том же процессе.

В нейро-нечетких моделях процесс выбора структуры включает в себя следующие основные варианты:

1. Выбор входных переменных. Это связано не только с физическими входами  $u$ , но и с динамическими регрессорами, определяемыми входными и выходными лагами,  $n_y$  и  $n_u$  соответственно.

Предварительные знания, понимание поведения процесса и цель моделирования являются типичными источниками информации для выбора исходного набора возможных входов. Автоматический выбор данных может быть использован для сравнения различных структур с точки зрения некоторых определенных критериев эффективности.

2. Количество и тип функций принадлежности, количество правил. Эти два структурных параметра взаимосвязаны (для большего количества функций принадлежности необходимо определить больше правил) и определить уровень детализации, называемый детализацией модели.

Цель моделирования и объем доступной информации (знаний и данных) будут определять этот выбор. Автоматизированные методы могут использоваться для добавления или удаления функций и правил принадлежности.

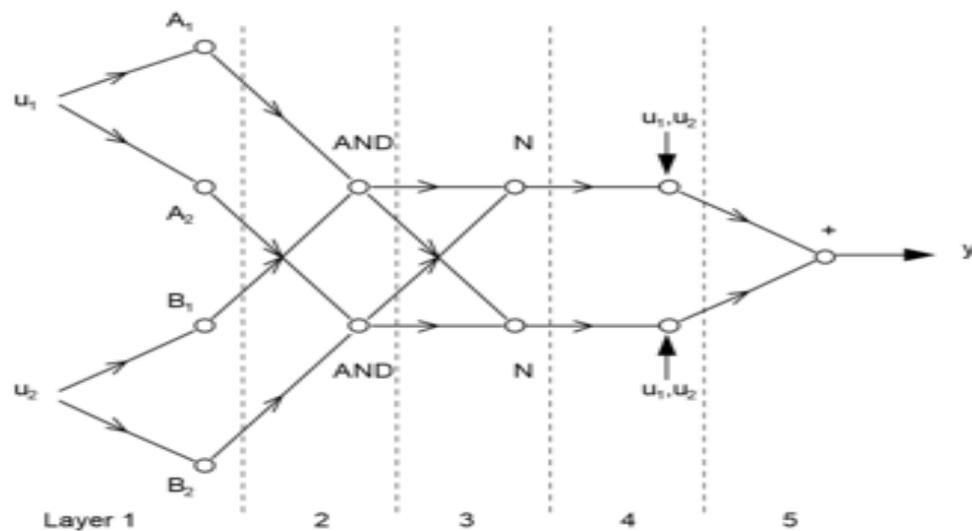


Рисунок 3.7. Структура сети ANFIS.

### Алгоритм обучения ANFIS.

Когда помещенные параметры фиксированы, общий выход представляет собой линейную комбинацию следующих параметров. В символах выход  $y$  можно записать в виде:

$$\begin{aligned}
 y &= \frac{\alpha_1}{\alpha_1 + \alpha_2} y_1 + \frac{\alpha_2}{\alpha_1 + \alpha_2} y_2 \\
 &= \bar{\alpha}_1 (c_{11} u_1 + c_{12} u_2 + c_{10}) + \bar{\alpha}_2 (c_{21} u_1 + c_{22} u_2 + c_{20}) \\
 &= (\bar{\alpha}_1 u_1) c_{11} + (\bar{\alpha}_1 u_2) c_{12} + \bar{\alpha}_1 c_{10} + (\bar{\alpha}_2 u_2) c_{21} + (\bar{\alpha}_2 u_2) c_{22} + \bar{\alpha}_2 c_{20}
 \end{aligned} \tag{3.6}$$

которая, линейна в следующих параметрах:  $c_{ij}$  ( $i = 1, 2; j = 0, 1, 2$ ) гибридный алгоритм корректирует последующие параметры  $c_{ij}$  в прямом проходе и помещенные параметры  $\{a_i, b_i, c_i\}$  в обратном проходе. В прямом проходе сетевые входы распространяются вперед до уровня 4, где последующие параметры идентифицируются методом наименьших квадратов. В обратном проходе сигналы ошибки распространяются назад, а помещенные параметры обновляются с помощью градиентного спуска.

Поскольку правила обновления для предпосылки и последующие параметры развязаны в гибридном правиле обучения, ускорение вычислений может быть возможным, используя варианты градиентного метода или другие методы оптимизации на исходных параметрах.

### **Оценка глобальных наименьших квадратов.**

Глобальный метод оценки наименьших квадратов дает параметры, которые минимизируют следующий критерий ошибки прогнозирования:

$$\theta = \arg \min \sum_{k=1}^N \left( y_k^* - \sum_{i=1}^K \gamma_i(\mathbf{x}_k) [\mathbf{x}_k^T \mathbf{1}] \theta_i \right)^2 \quad (3.7)$$

где  $\theta^T = [\theta_1^T \dots \theta_K^T]$  является конкатенацией всех индивидуальных правил векторных параметров. Для матриц данных этот критерий можно переписать в матричной форме:

$$\theta = \arg \min (\mathbf{y}^* - \Lambda \theta)^T (\mathbf{y}^* - \Lambda \theta) \quad (3.8)$$

$$\Lambda = [\Gamma_1 \varphi \dots \Gamma_K \varphi],$$

где  $\varphi = [\mathbf{X} \mathbf{1}]$  и  $\Gamma_i = \text{diag}(\gamma_i(\mathbf{x}_1) \dots \gamma_i(\mathbf{x}_N))$ , т. е. диагональная матрица, имеющая  $\gamma_i(\mathbf{x}_k)$  в качестве ее  $k$ -го диагонального элемента. Тогда оптимальное решение получается непосредственно с использованием матричного псевдообратного:

$$\theta = (\Lambda^T \Lambda)^{-1} \Lambda^T y^* \quad (3.9)$$

### Оценка локальных наименьших квадратов.

Хотя глобальное решение дает минимальную ошибку прогнозирования, оно может смещать оценки последствий в качестве параметров локальных моделей. Если требуются локально соответствующие параметры модели, следует использовать взвешенный подход наименьших квадратов к каждому правилу. Это делается путем минимизации множеств  $K$ -взвешенных локальных критериев LS:

$$\theta_i = \arg \min (y^* - \varphi \theta_i)^T \Gamma_i (y^* - \varphi \theta_i), \quad i = 1, 2, \dots, K \quad (3.10)$$

для которых решением является:

$$\theta_i = (\varphi^T \Gamma_i \varphi)^{-1} \varphi^T \Gamma_i y^*, \quad i = 1, 2, \dots, K \quad (3.11)$$

В этом случае последующие индивидуальные правила параметров оцениваются, независимо друг от друга, и поэтому результат не зависит от взаимодействия правил. Однако, в то же время, получается большая ошибка прогнозирования, чем с глобальными наименьшими квадратами.

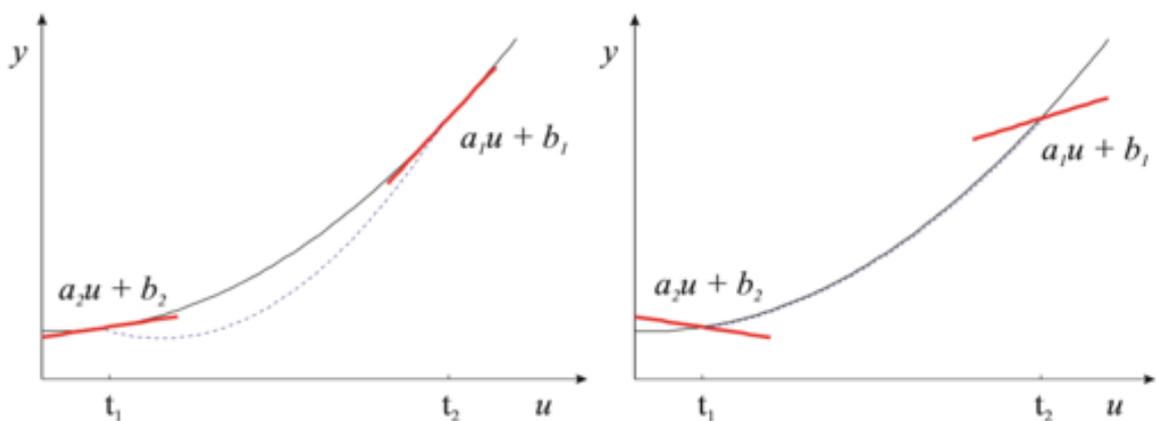


Рисунок 3.8. Результаты локальной (левой) и глобальной (правой) оценки соответствующих параметров. Пунктирная линия - это выход модели.

При интерпретации моделей ANFIS, полученных из данных, необходимо знать компромиссы между локальной и глобальной оценкой. Ограниченная и многокритериальная оптимизация также может применяться для ограничения независимости в параметрах.

### **Ограниченная оценка.**

Знания о динамической системе, такие как ее стабильность, минимальное или максимальное статическое усиление или время ее установления, могут быть переведены в выпуклые ограничения на последующие параметры. Используя данные ввода-вывода, оптимальные значения параметров затем определяются квадратичным программированием, а не наименьшими квадратами.

Существует два типа ограничений: глобальный и локальный. Локальные ограничения представляют собой подробные знания, относящиеся к каждому конкретному правилу, тогда как глобальные ограничения относятся ко всей модели и поэтому должны ссылаться на некоторые глобальные системные свойства, такие как общая стабильность.

Чтобы убедиться в этом, поймем, что возможную TS-модель можно рассматривать как одну квазилинейную систему

$$y = \left( \sum_{i=1}^K \gamma_i(\mathbf{x}) \mathbf{a}_i^T \right) \mathbf{x} + \sum_{i=1}^K \gamma_i(\mathbf{x}) b_i = \mathbf{a}^T(\mathbf{x}) \mathbf{x} + b(\mathbf{x}) \quad (3.12)$$

с входными зависимыми «параметрами»  $a(x)$ ,  $b(x)$ , которые являются выпуклыми линейными комбинациями индивидуальных последовательных параметров  $a_i$  и  $b_i$ , то есть:

$$\mathbf{a}(\mathbf{x}) = \sum_{i=1}^K \gamma_i(\mathbf{x}) \mathbf{a}_i, \quad b(\mathbf{x}) = \sum_{i=1}^K \gamma_i(\mathbf{x}) b_i. \quad (3.13)$$

Это свойство позволяет нам определять глобальные выпуклые ограничения для всей модели.

## **Ошибка обратного распространения (Error Backpropagation).**

Backpropagation - это процедура с градиентным спусками (включая все сильные и слабые стороны градиентного спуска) с функцией ошибки, которая получает все  $n$  весов в качестве аргументов и передает сигнал об ошибке, т.е. будучи  $n$ -мерным. На функции ошибки, точка малой ошибки или даже точка наименьшей ошибки запрашивается с помощью градиентного спуска.

Разность между желаемым выходом  $y^*$  и выходом сети  $y$  называется ошибкой. Эта ошибка используется для настройки веса в сети путем минимизации следующей функции затрат:

$$J = \frac{1}{2} \sum_{k=1}^N e_k^2 \quad e_k = y_k^* - y_k \quad (3.14)$$

## **Инициализация антецедента функций принадлежности**

Для успешного применения градиентного спуска к параметрам функции принадлежности важна хорошая инициализация. Несколько методов инициализации рассмотрены в этом разделе.

### **Функции принадлежности на основе шаблонов.**

С помощью этого метода домены предшествующих переменных являются предварительно выделенными рядом функций принадлежности. Они обычно равномерно распределены и имеют форму. Затем создается база правил, охватывающая все комбинации антецедентных терминов.

Суровым недостатком такого подхода является то, что число правил в модели растет экспоненциально. Кроме того, если не известно, какие переменные вызывают нелинейность системы, все предшествующие переменные обычно распределяются равномерно. Однако сложность поведения системы обычно неравномерна. Некоторые рабочие области могут быть хорошо аппроксимированы локальной линейной моделью, в то время как другие регионы требуют довольно тонкого разбиения.

Чтобы получить эффективное представление с минимальным количеством правил, функции принадлежности должны быть размещены так, чтобы они фиксировали неоднородное поведение системы.

### Дискретные методы поиска.

Итерационные алгоритмы поиска дерева могут быть применены для декомпозиции antecedentного пространства на гипер-прямоугольники с помощью ортогональных осколков. На каждой итерации область с наихудшей локальной ошибкой ошибки делится на две половины (или другие части). Выбираются разрывы во всех измерениях входного сигнала и выбирается тот, который имеет наибольшее улучшение производительности.

Это последовательное разделение прекращается, когда достигается конкретная цель ошибки или когда достигается желаемое количество правил.

Преимуществом такого подхода является его эффективность для высокоразмерных данных и прозрачности полученного раздела.

Недостатком является то, что процедура построения дерева является субоптимальной (жадной), и, следовательно, количество полученных правил может быть довольно большим.

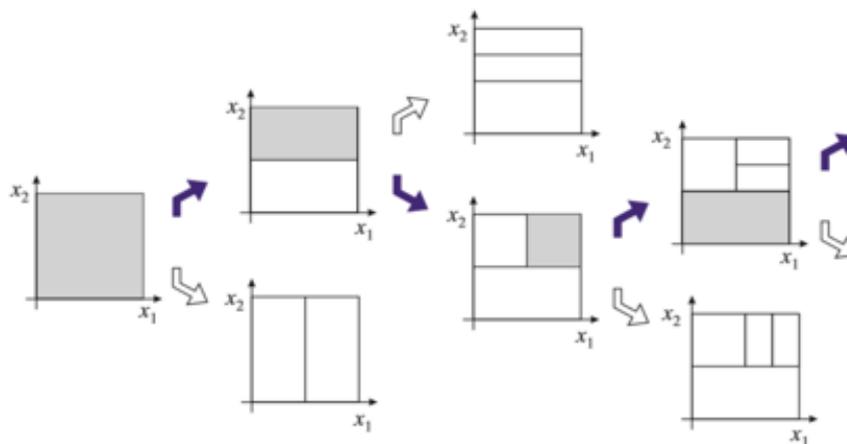


Рисунок 3.9. Антецедентное разложение пространства с помощью алгоритма эвристического поиска. Темные области представляют собой правила с наихудшим локальным соответствием на данном этапе.

## Нечеткая кластеризация (Fuzzy Clustering).

Методы построения, основанные на нечеткой кластеризации, исходят из анализа данных и распознавания образов, где понятие нечеткой принадлежности используется для представления степени, в которой данные объекта похожи на какой-то прототипный объект. Степень сходства может быть рассчитана с использованием подходящей меры расстояния. Основываясь на сходстве, векторы данных группируются таким образом, что данные в кластере как можно более похожи, а данные из разных кластеров настолько разнообразны, насколько это возможно.

На рисунке 3.10 приведен пример двух кластеров в  $R^2$  с прототипами  $v_1$  и  $v_2$ . Разделение данных выражается в *матрице нечеткого разбиения*  $U = [\mu_{ij}]$ , элементами которой являются принадлежность степени векторов данных  $x_k$  в нечетких кластерах с прототипами  $v_j$ .

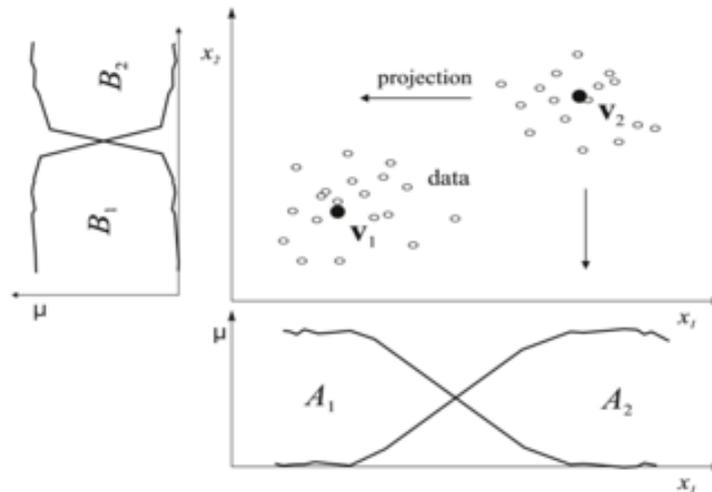


Рисунок 3.10. Идентификация функций принадлежности через нечеткую кластеризацию.

Затем antecedentesные функции принадлежности извлекаются путем проецирования кластеров на отдельные переменные. Для инициализации моделей ANFIS первого порядка прототипы могут быть определены как линейные подпространства, или кластеры являются эллипсоидами с

адаптивно определенной формой. Количество кластеров в данных может быть определено как приоритет, который запрашивается автоматически, используя критерии валидности кластеров и методы слияния.

### **Пример моделирования (Статическая функция)**

В этом разделе приводится пример моделирования, иллюстрирующие несколько важных вопросов, связанных с обучением нейро-нечетких систем. Пример простой задачи о неподвижной статической функции. Он демонстрирует типичную процедуру построения нейро-нечеткой модели.

Численные результаты показывают, что улучшение производительности достигается за счет получения правил if-then, которые не являются полностью релевантными как локальное описание системы. Опишем одномерную функцию  $y = \sin(u)$  моделью ANFIS с линейными последующими функциями. Мы выбираем количество правил, которых должно быть пять, и построим исходную модель путем кластеризации данных  $U \times Y$ , используя методологию, основанную на алгоритме Густафсона-Кесселя. Получены следующие правила:

Если  $u$  равно A1, то  $y = 5.721u + 0.030$

Если  $u$  равно A2, то  $y = 0.035u + 0.904$

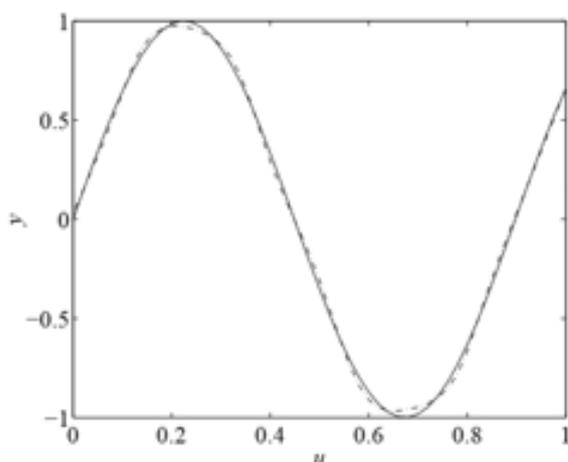
Если  $u$  есть A3, то  $y = -5.302u + 2.380$

Если  $u$  равно A4, тогда  $y = 0.734u - 1.413$

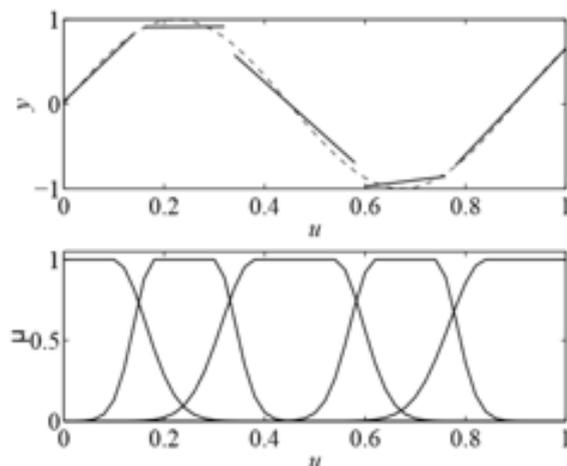
Если  $u$  есть A5, то  $y = 6.283u - 5.623$

Функция этой исходной модели показана на рисунке 3.11а. Функции принадлежности и соответствующие локальные модели приведены на рисунке 3.11б.

Функции принадлежности обозначаются слева направо от A1 до A5.



3.11a Данные (сплошные) и модели (пунктирные).



3.11b Локальные модели (сверху) и функции принадлежности (внизу)

### Оценка кредитного риска с помощью ANFIS.

Рассмотрим моделирование с помощью ANFIS-модели, такого слабоструктурированного процесса, как оценка кредитного риска. Для моделирования используется программное обеспечение MATLAB для создания предиктора ANFIS. Процедура и обучение ANFIS реализуются с использованием стандартного нечеткого инструментария в программном обеспечении MATLAB.

Кредит является основным продуктом, который банки предлагают своим клиентам, который генерирует большую часть прибыли, но в то же время вызывает наибольшие потери в случае дефолта.

Кредитный риск – это риск потерь из-за дефолта заемщика. Простое увеличение вероятности риска может негативно повлиять на стоимость портфеля. Вот почему решение о предоставлении или отказе от кредита является ключевым решением для банка. Кредитный риск представляет собой сложную проблему измерения. Простое знание выдающегося кредита недостаточно для оценки его риска. Возможные убытки зависят от рассмотрения обязательств, а также от вероятности дефолта контрагентов, что невозможно легко измерить. Этот раздел содержит описание используемых наборов данных и предоставляет предварительную обработку данных.

Используемый набор данных взят из банка. Множество содержит непрерывные переменные, такие как возраст, стаж работы, способность погашения, которые составляют входные данные ANFIS. Собранные конфиденциальные данные, связаны с погашением кредита, 108-ми клиентов в банке. Некоторые из этих клиентов откладывают погашение своих ссуд и считаются клиентом с высоким риском.

Таблица 3.1. Описание переменного «Возраст».

Возраст	Число	Процентное соотношение
< 30 лет	23	21,29%
30 лет <= and < 60 лет	81	75%
>= 60 лет	4	3,71%
Всего	108	100%

Таблица 3.2. Описание переменного «Стаж работы».

Стаж работы	Число	Процентное соотношение
< 2 года	6	5,55%
2 года <= and < 10 лет	64	59,26%
>= 10 лет	38	35,19%
Всего	108	100%

Таблица 3.3. Описание переменного «Способность погашения».

Способность погашения	Число	Процентное соотношение
< 200 man	7	6,5%
200 <= and < 400 man	55	50,9%
>= 400 man	46	42,6%
Всего	108	100%

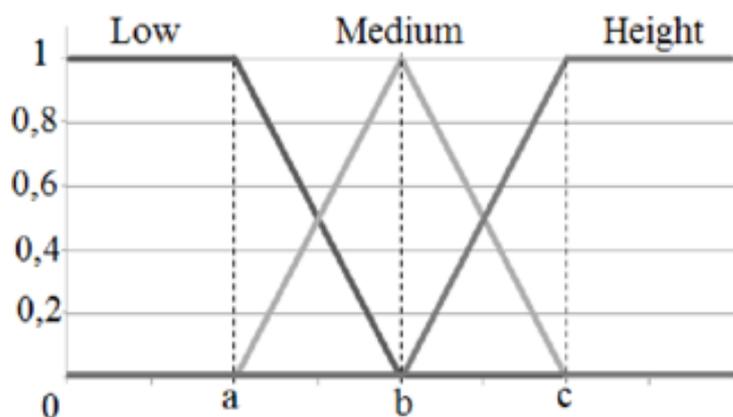


Рисунок 3.12. График функции принадлежности ANFIS.

Таблица 3.4. Значение использованных переменных в графике функции принадлежности.

Переменные	Низкий	Средний	Высокий
Возраст	30	45	60
Стаж работы	2	6	10
Способность погашение	200	300	400

Нечеткая система имеет три входных переменных. Каждая переменная имеет три функции принадлежности.

Возможные комбинации принадлежности затем достигают 27. При запуске процесса обучения ANFIS автоматически вычисляет эти комбинации, чтобы вывести правила нечеткого вывода.

Основываясь на интервью с экспертами банка, среди 108 клиентов 51 клиентов считаются клиентами высокого риска. Рискованный клиент назначил оценку, указывающую на вероятность задержки в погашении долгов.

Целью модели оценки кредита является прогнозирование этой оценки для каждого клиента на основе некоторых объясняющих переменных.

В качестве выхода используется DelProb (delay probability – вероятность задержки). DelProb находится отношением количеств отложенных платежей на общее количество платежей.

Результаты ANFIS для прогнозируемого кредитного скоринга по сравнению с фактическими вероятностями задержки показаны на рисунке 3.13.

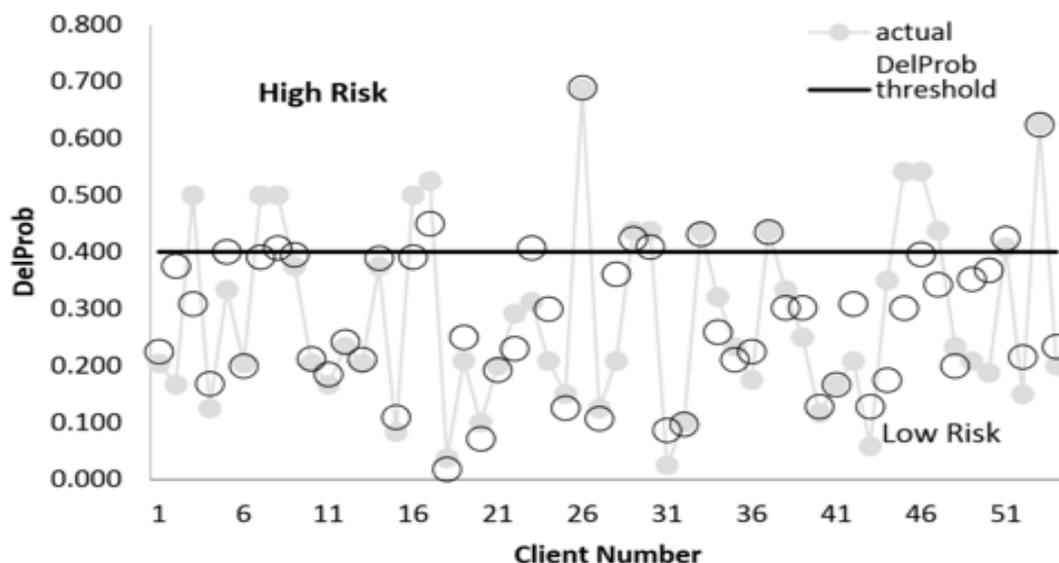


Рисунок 3.13. Результаты ANFIS для кредитного скоринга.

Кредитный скоринг (рейтинг) обычно связан с оценкой потенциальных рисков, соответствующих предоставлению баллов. Для сравнения результатов оценки скоринга методом ANFIS используется средняя абсолютная процентная погрешность (MAPE):

$$MAPE = \frac{1}{N} \sum_{i=1}^N \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100\% \quad (3.15)$$

где  $y_i$  и  $\hat{y}_i$  представляют фактические и предполагаемые вероятности задержки  $i$ -го клиента соответственно, а  $N$  - количество клиентов.

Как видно, предиктор ANFIS имеет удовлетворительные результаты для клиентов с низким уровнем риска, однако его результаты для клиентов с высоким уровнем риска не очень хорошие и имеют относительно больше ошибок.

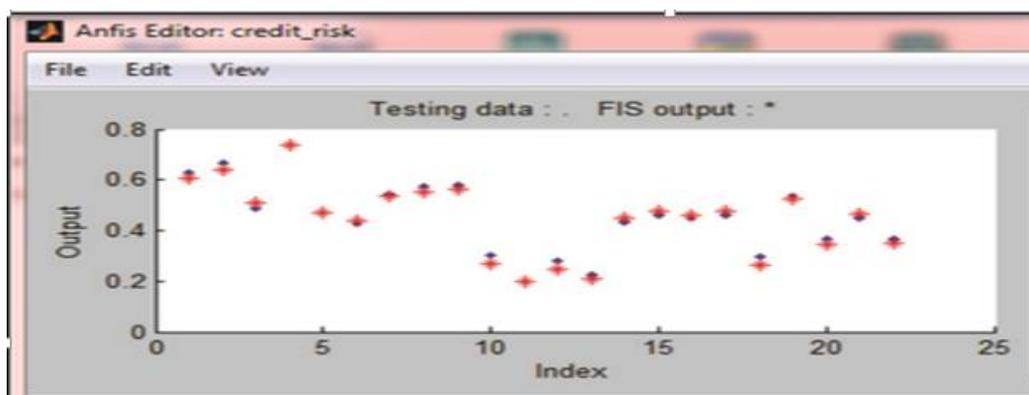


Рисунок 3.14. Вывод кредитного риска для тестирования данных.

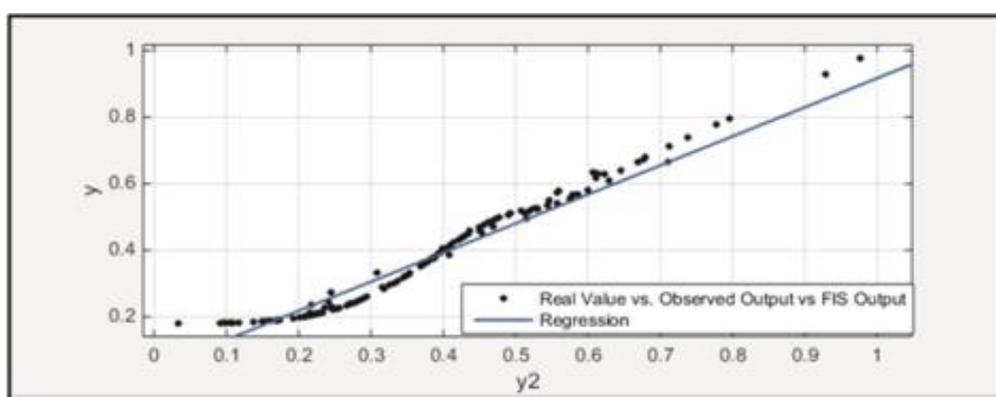


Рисунок 3.15. Сравнение результатов кредитного риска ANFIS с реальными данными на этапе обучения.

После завершения учебного процесса и этапов проверки внешней системы, тестированные данные могут быть введены в систему без представления реальных результатов, чтобы модель могла предсказать запрошенные значения.

Однако следующие диаграммы показывают, что выходные значения были предсказаны MATLAB в дополнение к пригодности реальных и прогнозируемых выходов по отношению к данным тестирования.

Следующие рисунки (3.16-3.18) показывают успех и высокую точность моделей ANFIS.

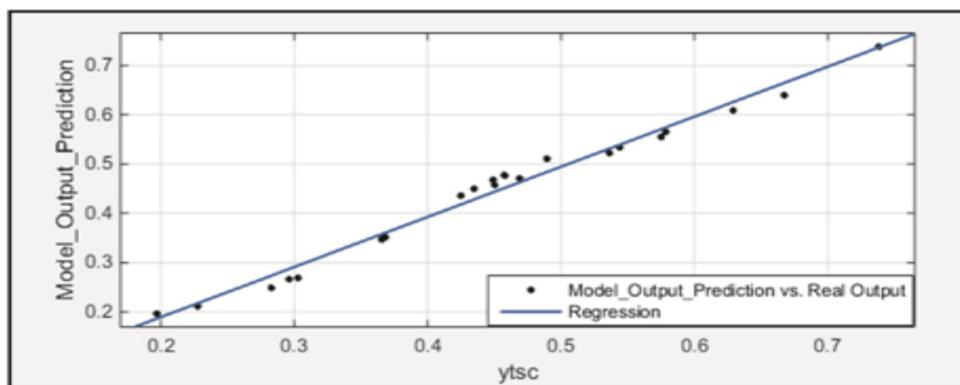


Рисунок 3.16. Сравнение результатов кредитного риска с реальными данными в ANFIS на этапе тестирования.

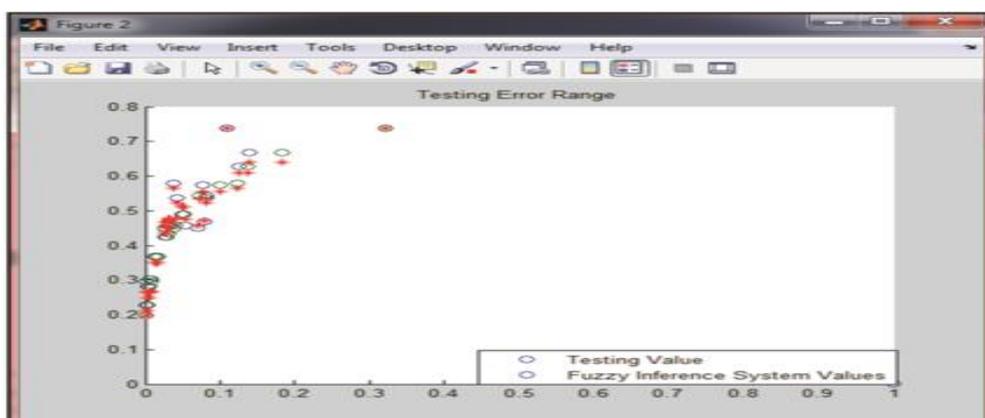


Рисунок 3.17. Отображение реальных значений и вычисленных значений кредитного риска в модели ANFIS для тестирования данных.

Real Outputs	System Data
7.541863	7.300358194
8.0020411	7.671326058
5.8690365	6.124778497
8.8507744	8.854441146
5.6218553	5.642688195
5.0954197	5.227879389
6.5226524	6.401113842
6.8951929	6.65319125
6.9363679	6.78416766
3.6318102	3.223890057
2.3611756	2.353280617
3.3923974	2.983418511
2.7301778	2.529286792
5.213519	5.392508438
5.4902332	5.713954966
5.3981194	5.486986412
5.4845735	5.727758581
3.5514244	3.195259187

Рисунок 3.18. Сравнение результатов модели оценки кредитного риска с реальными результатами.

Выводы показывают, что применение нейро-нечеткой модели ANFIS, является хорошо применимым к оцениванию кредитного риска с более точными результатами.

## ВЫВОДЫ И ПРЕДЛОЖЕНИЯ

В этой диссертационной работе представляется применение нейро-нечеткого моделирования к слабоструктурированным процессам, моделью ANFIS в среде MATLAB. Выводы на основе моделирования кредитных рисков показывают, что применение нейро-нечеткого моделирования с помощью ANFIS, предоставляет точное моделирование слабоструктурированного процесса.

Преимуществом нейро-нечеткого моделирования является то, что оно представляет собой гибкую структуру, в которой можно сочетать различные парадигмы, обеспечивая, с одной стороны, прозрачный интерфейс с дизайнером и, с другой стороны, инструмент для точного нелинейного моделирования и управления. Основанный на правилах, характер нейро-нечетких моделей позволяет провести анализ и интерпретации результата. Обычные методы количественной валидации могут дополняться человеческим опытом, который часто включает в себя эвристические знания и интуицию.

Недостатком нейро-нечеткого моделирования является то, что современные методы построения и настройки нечетких моделей довольно сложны, и их использование требует специальных навыков и знаний. В этом смысле он, вероятно, никогда не станет «одной кнопкой», полностью автоматизированной методикой идентификации. Нейро-нечеткое моделирование скорее следует рассматривать как интерактивный метод, облегчающий активное участие пользователя в сеансе моделирования с помощью компьютера. Это в определенной степени относится и к другим, более известным методам. Моделирование сложных систем всегда будет интерактивным.

## СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. А. Бахраммирзаи: Сравнительный обзор приложений искусственного интеллекта в области финансов: искусственные нейронные сети, экспертная система и гибридные интеллектуальные системы, Нейронные вычисления и приложения, 1165–1195, 2010.
2. А.Л. Заде: Нечеткая логика вычислений со словами, IEEE транз. на нечеткие системы, изд. 4, стр.103-111, 1996.
3. А.Л. Заде: Нечеткая логика – как логика естественных языков. Анализ и проектирование интеллектуальных систем с использованием методов мягких вычислений (стр. 1-2), Берлин, Гейдельберг, 2007.
4. Барто, Р. Саттон и К. Андерсон: Нейрон-подобные адаптивные элементы, которые могут решать сложные проблемы управления обучением. IEEE-транзакции по системам, человеку и кибернетике: 834–846, 1983.
5. Бездек Ж.К., 1993. Нечеткие модели - что они и почему - редакционные. IEEE Транзакции на нечеткие системы, том 1, стр. 1-5. Кокс, Э., 1994. Справочник по нечетким системам. Кембридж, Массачусетс: Академическая пресса. Дюбуа Д. и Праде Х., 1980. Нечеткие множества и системы: теория и приложения. Нью-Йорк: Академическая пресса.
6. Б. Коско: Нейронные сети и нечеткие системы, 1991. Нечеткие системы как универсальные аппроксиматоры. Материалы Международной конференции IEEE по нечетким системам. Сан-Диего, Калифорния, стр. 1151-1162, 1992.
7. Гочич М., Мотамеди С., Шамширбанд С., Петкович Д., Хашим Р.: Потенциал адаптивной нейро-нечеткой системы вывода для оценки показателей засухи. Стохастические исследования окружающей среды и оценка рисков, 2010.
8. Г. Рот и У. Дикке. Эволюция мозга и интеллекта. Тенденции в когнитивных науках: 250-257, 2005.

9. Гоффманн Ф., Баессен Б., Мартенс Дж., Пут Ф., Вантениен ж. (2002). «Сравнение генетического нечеткого и нейро-нечеткого классификатора для кредитного скоринга». Международный журнал интеллектуальных систем, 17, 1067- 1083.
10. М. Ахбари: Применение систем нейрон-нечетких рассуждений в кредитных кредитах кредитных организаций корпоративных клиентов, 2010.
11. М. Гудзари, М. П. Фрейтас, MIA-QSAR связанный с адаптивными системами нейро-нечетких выводов (PCA-ANFIS) для анализа активности анти-ВИЧ обратной транскриптазы производных ТИВО. Европейский журнал медицинской химии: 1352-1358, 2010.
12. М.Э. Хасселмо, Ж.Л. МакКлинд: Нейронные модели памяти // Современное мнение в нейробиологии. изд. 9. Р. 184-188, 1999.
13. М. Моусави, Ж. Кевин По и Х. Сирфиан По: Нечеткая экспертная система для управления рисками проектов. Пятая международная конференция по управлению проектами, 2009 год.
14. Николич В., Петкович Д., Пор Л., Шамширбанд С., Замани М., Шойбашич Ж. и Мотамеди С.: Потенциал нейро-нечеткой методологии для оценки уровня шума ветровых турбин. Механические системы и обработка сигналов: стр. 715-722, 2016.
15. Р. Бабушка. Нечеткое моделирование для контроля. Kluwer Academic Publishers, Бостон, США, 1998.
16. С.Р. Янг, С.-Т. Сан и Э. Мизутани: Нейро-нечеткие и мягкие вычисления; вычислительный подход к обучению и машинной разведке, 1997.
17. Хелендоорн и Д. Дрианков: Идентификация нечеткой модели: выбранные подходы. Берлин, Германия, 1997.
18. Эбрат М., Годси Р. : Оценка риска проекта строительства с использованием системы нечетких выводов с адаптивной сетью: эмпирическое исследование, 1213-1227, 2014.

19. Э.Р. Кандэл, Ж.Х. Шварц, Т.М. Джессель. Принципы нейронной науки, 2000.
20. Abramova M.I.: Informatics- Belgorod: NRU Belgorod State University, 2011.
21. Babak V.F., Rijkenko I.N.: Perfection of the methodology of planning information systems. Web: <http://emanual.ru/download/1638.html>
22. Dushin V.K.: Theoretical foundations of information processes and systems, 2009.
23. Kalyanov G.N.: Modeling, analysis, reorganization and automation of business processes. – М.: Finance and Statistics, 2007.
24. Kudryavtsev E.M.: GPSS World. Fundamentals of imitation modeling of various systems, 2004. – 320 p.
25. Korolev A.L.: Computer modelling. – М.: Binomial: Knowledge laboratory, 2010.
26. Sovetov B.Y., Yakovlev S.A.: System modeling, 2001. – 343 p.
27. Sovetov B.Y., Yakovlev S.A.: System modeling. – St. Petersburg State Electrotechnical University. -7th edition, 2012. – 342 p.
28. Веб-источники: <https://www.computer.org>; [www.mathworks.com](http://www.mathworks.com)

## RESUME

In this dissertational work, is submitted application of neural-fuzzy modeling to weakly structured processes, the ANFIS model in the MATLAB environment, appears. Conclusions on the basis of modeling credit risk, the use of neural-fuzzy modeling with the help of ANFIS, provides an accurate modelling of a weakly-structured process.

The advantage of neural-fuzzy modeling is that it is a versatile structure in which you can combine various paradigms, assuring, on the one hand, a transparent interface with the designer and, on the other hand, a tool for accurate nonlinear modeling and control. Based on the rules, the nature of neuro-fuzzy models allows for analysis and interpretation of the result. The usual methods of quantitative validation can added by human experience, which often includes heuristic knowledge and intuition.

The disadvantage of neural-fuzzy modeling is that modern methods of formation and settings fuzzy models are pretty difficult, and their use request special skills and knowledge. In this sense, it probably will never become a "one button", a fully automated identification technique. Neuro-fuzzy modeling should pretty be seen as an interactive method that facilitates the active involvement of the user in a simulation session using a computer. This to a certain extent applies to other, more well-known methods. Modeling of compound systems will always be interactive.