Леденева Т.М., Татаркин Д.С., Тарасова А.С.

Основы нечеткого моделирования в среде MatLab

Учебное пособие

Воронеж 2006

УДК 681.3.068

Печатается по решению научно-методического совета факультета прикладной математики, информатики и механики ВГУ, протокол №5 от 20.02.2006.

Леденева Т.М., Татаркин Д.С., Тарасова А.С. Основы нечеткого моделирования в среде *MatLab*: Учеб. пособие. Воронеж: Издательско-полиграфический центр ВГУ, 2006. 51 с.

Рецензент: д.т.н., профессор Лозгачев Г.И.

В учебном пособии рассматривается комплекс взаимосвязанных задач нечеткого моделирования: системы нечеткого логического вывода (в том числе аппроксимация нечетких систем нелинейных систем продукционными правилами), решение нечетких реляционных уравнений и нечеткая кластеризация. Все эти задачи составляют основу нечетких моделей управления. В теоретической части учебного пособия кратко излагаются основные понятия и методы решения перечисленных задач. Вторая часть содержит лабораторный практикум в среде *MatLab*.

Учебное пособие предназначено для студентов 3-4 курсов дневного отделения специальности 010501 (010200) – Прикладная математика и информатика, 010503 (351500) – Математическое обеспечение и администрирование информационных систем и (351400) – Прикладная математика в юриспруденции, изучающих следующие дисциплины: «Обработка нечеткой информации», «Модели и методы принятия решений», «Методы экспертных оценок».

1. ОСНОВНЫЕ ПОНЯТИЯ НЕЧЕТКОГО МОДЕЛИРОВАНИЯ

Нечеткая система – это система, для описания которой используется аппарат теории нечетких множеств и нечеткая логика, при этом различают следующие способы такого описания:

– нечеткая спецификация параметров системы (функционирование системы может быть описано алгебраическим или дифференциальным уравнением, в котором параметры являются нечеткими числами);
– нечеткое (лингвистическое) описание входных и выходных переменных системы, которое обусловлено неточной информацией, получаемой от ненадежных датчиков, или качественной информацией, получаемой от эксперта;
– нечеткое описание системы в виде совокупности ***если-то*** **-** правил, отражающих особенности функционирования на качественном уровне.

Нечеткая система может обладать одновременно всеми перечисленными атрибутами. Нечеткие системы используются для моделирования, анализа данных, прогноза или управления.

Нечеткая система управления (НСУ) – это интеллектуальная система, использующая нечеткое описание управляемого процесса и системы его управления в виде базы нечетких правил для генерации последовательности управляющих решений, обеспечивающих достижение целей управления. Основой для построения НСУ является схема управления с участием эксперта, который на основе опыта и знаний об управлении объектом формирует описание процесса управления. Затем это описание преобразуется в базу правил и в дальнейшем используется в системе управления уже без участия эксперта. Идея нечеткого управления заключается именно в подражании действиям опытного человека-оператора. Нечеткие правила – это нечеткие продукционные правила[1], которые при фиксированной цели управления (например, сохранение значений управляемого параметра в некоторой области допустимых значений) описывают его стратегии на качественном уровне. Приведем фрагмент базы правил для некоторой системы управления:

если x *мало* и y *мало*, ***то*** z *слабо увеличить*;
если x *мало* и y *велико*, ***то*** z *немного уменьшить*;
если y *не очень велико*, ***то*** z *не изменять*;
если y *очень велико*, ***то*** стоп.

---

[1] Под *продукцией* понимается кортеж следующего вида:

$$<(i);Q;P;A \rightarrow B;N>,$$

где i – имя продукции, Q характеризует сферу применения продукции, A®B – ядро продукции (обычное прочтение ядра выглядит как: *если* A, *то* B), P – условие применимости ядра продукции или предусловие, N – постусловие продукции описывает действия и процедуры, которые необходимо выполнить после реализации B. Основной частью продукции является ядро, остальные элементы носят вспомогательный характер, поэтому в наиболее простом виде продукция может состоять лишь из имени и ядра.

Каждое правило представляет собой нечеткую инструкцию, записанную на естественном (возможно формализованном) языке, а в целом 1)-4) – это нечеткий алгоритм управления.

Алгоритмы нечеткого управления используются в *нечетких контроллерах*, под которыми подразумеваются программно-аппаратные системы, управляющие некоторым процессом. Такого рода системы имеют огромное число приложений.

Пусть X и Y – множество значений входной и выходной переменных соответственно. Функционирование системы описывается функцией $f : X \to Y$. Для систем управления функция f называется *функцией управления*. Нечеткие системы можно рассматривать как обобщение интервальных систем, являющихся в свою очередь обобщением четких систем. На рис. 1.1 приведен пример четкой функции f и ее интервальной и нечеткой форм.

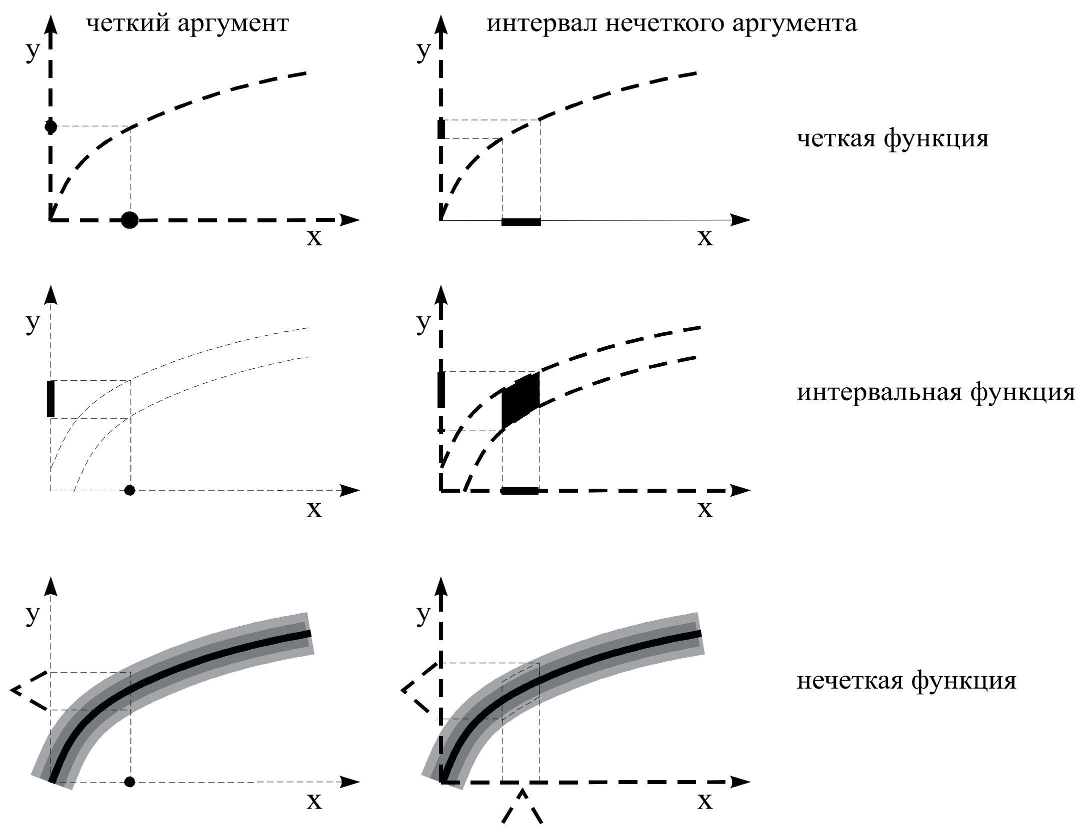

Рис. 1.1

Важнейший этап моделирования сложных систем – восстановление неизвестной функции f, после чего для каждого конкретного значения входной переменной $x \in X$ становится возможным найти конкретное значение выходной переменной $y \in Y$. Математический аппарат, использующийся для описания системы, предполагает свой арсенал методов для решения этой задачи. В случае нечеткого описания системы нечеткими могут быть не только входы и выходы, но и функция f, являясь подмножеством декартова произведения $X \times Y$, может рассматриваться как нечеткое отношение.

Рассмотрим основные приемы нечеткого моделирования систем. Для описания объектов и явлений в условиях неопределенности используются понятия лингвистической и нечеткой переменной.

Нечеткая переменная задается тройкой

$$<a, U, A>,$$

где a – название переменной, U – универсальное множество (область определения a), A – *нечеткое множество*[2] на U с функцией принадлежности $m_A : U \to [0,1]$, описывающее ограничения на значение нечеткой переменной.

Лингвистическая переменная задается кортежем

$$<b, T, U, G, M>,$$

где b – название переменной; T – *базовое терм-множество* или множество основных лингвистических значений (термов) переменной b, причем каждому из них соответствует нечеткая переменная с функцией принадлежности, заданной на U; G – *синтаксическое правило*, описывающее процесс образования из множества T новых значений лингвистической переменной ($T^* = T \cup G(T)$ называется *расширенным терм-множеством*); M – *семантическое правило*, согласно которому новые значения лингвистической переменной b, полученные с помощью G, отображаются в нечеткие переменные (это может быть процедура экспертного опроса, позволяющая приписать каждому новому значению, образованному процедурой G, некоторую семантику путем формирования соответствующего нечеткого множества). Шкала U, на которой определена лингвистическая переменная, может быть числовой или нечисловой.

Нечеткое число – это нечеткая переменная, определенная на числовой оси функцией принадлежности $m_A : \mathrm{P} \to [0,1]$. Среди нечетких чисел самые простые - треугольные и трапециевидные.

Треугольным нечетким числом A с центром в точке a, левой шириной $l > 0$ и правой шириной $r > 0$ называется нечеткое множество A с функцией принадлежности вида

$$m_A(x) = \begin{cases} 1 - (a - x)/l, & \text{если } a - l \le x \le a; \\ 1 - (x - a)/r, & \text{если } a \le x \le a + r; \\ 0, & \text{иначе.} \end{cases}$$

Трапециевидное нечеткое число A с отрезком толерантности $[a,b]$, левой шириной $l > 0$ и правой шириной $r > 0$ задается функцией

---

[2] Пусть U есть множество, счетное или нет, и $x \in U$. *Нечетким подмножеством* A *множества* U называется множество упорядоченных пар

$$A = \{(x, m_A(x))\},$$

где $m_A(x)$ – *функция принадлежности*, принимающая свои значения в $[0,1]$ (в общем случае в линейно упорядоченном множестве M) и определяющая степень принадлежности элемента x к подмножеству A.

$$m_A(x)=\begin{cases} 1-(a-x)/l, & если\ a-l\le x\le a; \\ 1, & если\ a\le x\le b; \\ 1-(x-a)/r, & если\ b\le x\le b+r; \\ 0, & иначе. \end{cases}$$

Треугольное нечеткое число с центром в точке *a* можно рассматривать как нечеткое значение высказывания *x приблизительно равен a*, в то время как трапециевидное число обозначает нечеткое значение высказывания *x находится приблизительно в интервале* $[a,b]$. Величины l и r также называются соответственно левым и правым коэффициентами нечеткости и показывают насколько неточно (нечетко) определены границы числа. Трапециевидное нечеткое число обозначается кортежем (a,b,l,r); при $a=b$ получим треугольное нечеткое числа (a,l,r).

Пример. Пусть стоимость выпускаемой продукции оценивается с помощью понятий *малая, средняя, высокая* и изменяется от 100 до 5000 рублей. Для формализации такого описания введем лингвистическую переменную *Стоимость,* которой соответствуют следующие элементы описания:

b = *Стоимость*;

T = {α_1 = *малая*, α_2 = *средняя*, α_3 = *высокая*};

$U = [100, 5000]$;

G = {X_1=*не*, X_2= *достаточно*} (тогда расширенное терм-множество может быть, например, таким T^* = {*малая, не высокая, средняя, достаточно высокая, высокая*};

$M = [\,m_{не\,a}(x)=1-m_a(x),\ m_{достаточно\,a}(x)=[m_a(x)]^{1,25}\,]$.

На рис. 1.2 каждому терму соответствует трапециевидное нечеткое число.

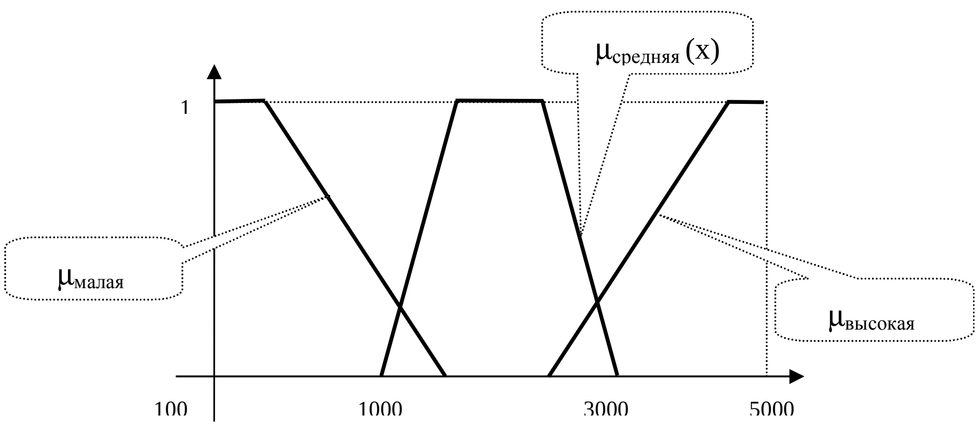

Рис. 1.2

Нечетким лингвистическим высказыванием будем называть высказывание одного из следующих типов:

– <u>*b есть a*</u>, где b – наименование лингвистической переменной, a – ее значение, которому соответствует некоторый лингвистический терм из базового терм-множества *T* лингвистической переменной b;

– *b есть X (a)*, где *X* – *модификатор*, соответствующий таким словам, как *очень, более или менее, почти, совсем* и другим, которые задаются процедурой *G*;
– составные высказывания, образованные из высказываний первых двух типов с помощью логических связок *и, или, не, если-то, тогда и только тогда*.

Составные нечеткие высказывания образуются одним из следующих способов:

а) путем соединения по принципу *и* или по принципу *или* нескольких элементарных высказываний, относящихся к одной и той же лингвистической переменной, например,

$$\underline{b \text{ есть } a_1} \ \textbf{и(или)} \ \underline{b \text{ есть } a_2}.$$

Заметим, что в этом случае путем равносильных преобразований составное высказывание можно упростить, приведя его к виду

$$\underline{b \text{ есть } a},$$

где $a = a_1$ ***и(или)*** a_2, т.е. функция принадлежности терма a образуется путем пересечения (объединения) функций принадлежности термов a_1 и a_2.

б) путем соединения по принципу *и* или по принципу *или* нескольких элементарных высказываний, относящихся к разным лингвистическим переменным, например,

$$\underline{b_1 \text{ есть } a_1} \ \textbf{и(или)} \ \underline{b_2 \text{ есть } a_2} \ \textbf{и(или)} \ \ldots \ \textbf{и(или)} \ \underline{b_N \text{ есть } a_N},$$

где $b_1, \ldots, b_N$ - лингвистические переменные, $a_1, \ldots, a_N$ – значения соответствующих лингвистических переменных. Логические связки *и* и *или* формализуются подходящими операциями над нечеткими множествами.

Целью *фазификации* является установление взаимно однозначного соответствия между конкретным числовым значением и лингвистическим значением некоторой переменной. Пусть a_i – числовое значение переменной b_i, a – некоторый терм с известной функцией принадлежности $m_a(x)$, тогда значение a_i используется в качестве аргумента функции принадлежности $m_a(x)$ и, тем самым, определяется числовое значение $b_i = m(a_i)$, которое и является результатом фазификации.

Нечеткое моделирование – это методология, согласно которой правило вида

$$R_i : \textbf{если } x \text{ есть } A_i, \textbf{ то } y \text{ есть } B_i \ \left(i = \overline{1,N}\right),$$

в котором *посылка (условие)* ***x есть A_i*** и *заключение* ***y есть B_i*** представляют собой нечеткие высказывания, переводится в четкую модель, т.е. функцию $f : X \to Y$, которая описывает связь между множествами *X* и *Y*. Совокупность нечетких правил $\{R_i\}$, предназначенных для формального представления эмпирических знаний или знаний экспертов в некоторой проблемной области, образует *базу правил*. Лингвистические переменные, которые используются в условиях нечетких правил, называются *входными переменными*, а в заключениях – *выходными*. При формировании базы правил необходимо определить множество входных лингвистических переменных, множество выходных лингвистических переменных и множество нечетких правил, связывающих входные переменные с выходными. Заметим, что нечеткие правила должны быть согласованы относительно используемых в них переменных. Согласованность правил означает, что в качестве условий и заключений правил могут использоваться только нечеткие лингвистические высказывания рассмотренных выше типов, и в каждом из нечетких высказываний должны быть определены функции принадлежности значений терм-множеств для каждой из лингвистических переменных.

Рассмотрим различные типы нечетких моделей.

Лингвистическая нечеткая модель имеет вид

$$R_i : \textbf{\textit{если}}\ x\ \textit{есть}\ A_i, \textbf{\textit{то}}\ y\ \textit{есть}\ B_i \quad \left(i=\overline{1,N}\right),$$

где x, y – входная и выходная лингвистические переменные; A_i, B_i – лингвистические термы (им соответствуют нечеткие переменные с определенными функциями принадлежности).

Каждое правило можно рассматривать как нечеткое отношение $R_i : X \times Y \to [0,1]$, которое вычисляется на основе нечеткой импликации, формализующей причинную связь между посылкой и заключением в виде $A_i \to B_i$. Существуют различные представления импликации. В самом простом и распространенном случае

$$R_i = A_i \times B_i \quad \Leftrightarrow \quad \mu_{R_i}(x,y) = \mu_{A_i}(x) \wedge \mu_{B_i}(y) = min\left(\mu_{A_i}(x), \mu_{B_i}(y)\right),$$

тогда нечеткое отношение R, соответствующее совокупности продукционных правил, определяется следующим образом:

$$R = \bigcup_{i=1}^{K} R_i, \quad \mu_R(x,y) = \max_i(\mu_{A_i}(x) \wedge \mu_{B_i}(y)) = \max_i min\left(\mu_{A_i}(x), \mu_{B_i}(y)\right).$$

Нечеткая реляционная модель может быть рассмотрена как расширение лингвистической модели, в которой отображение между входными и выходными нечеткими множествами представляется нечеткими отношениями. Введем $A_j = \left\{A_{jk} / k = \overline{1,N_j}, j = \overline{1,p}\right\}$ и $B = \left\{B_k / k = \overline{1,M}\right\}$ – множество лингвистических термов, определенных для переменных посылок и переменных заключений соответственно. База правил может быть представлена как нечеткое отношение между лингвистическими термами посылок и заключений $R : A_1 \times A_2 \times \ldots \times A_n \times B \to [0,1]$. В этой модели каждое правило содержит все возможные термы следствий, причем со своим весовым коэффициентом, заданным соответствующим элементом нечеткого отношения. С помощью этого веса можно легко настраивать модель. Нечеткая реляционная модель с одним входом и одним выходом показана на рис. 1.3.

Модель *Takagi – Sugeno (TS-модель)* можно рассматривать как комбинацию лингвистической и регрессионной моделей. Она задается в следующем виде:

$$R_i : \textbf{\textit{если}}\ x\ \textit{есть}\ A_i, \textbf{\textit{то}}\ y_i = f_i(x) \quad \left(i=\overline{1,N}\right).$$

В данной модели посылка представляет собой нечеткое высказывание, а заключение – четкую функцию. Функции f_i обычно имеют одинаковую структуру, только параметры для каждого правила различны.

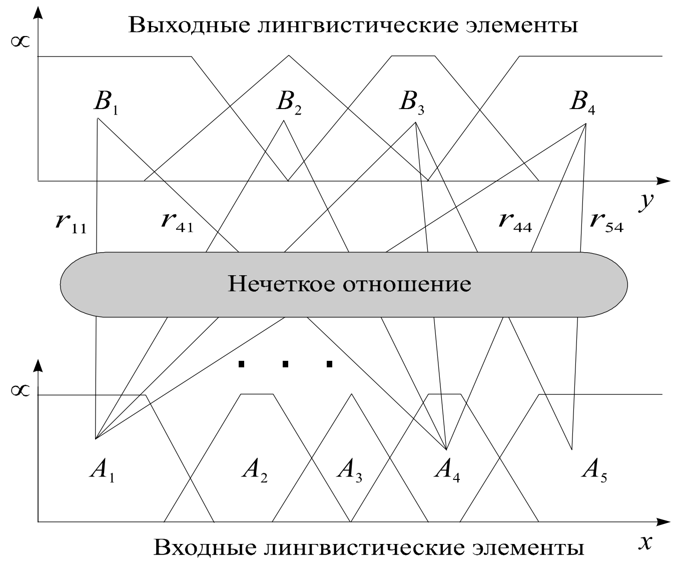

Рис. 1.3

Наиболее простой и полезной для приложений является линейная функция, так что правила будут иметь вид

$$R_i : \boldsymbol{если}\ x\ есть\ A_i, \boldsymbol{то}\ y_i = a_i x + b_i \quad \left(i = \overline{1,N}\right),$$

где a_i и b_i – параметры (в данной случае предполагается единственная входная переменная, если входных переменных несколько, то a_i – вектор параметров).

На рис. 1.4 представлена нечеткая *TS*-модель, которую можно рассматривать как кусочно-линейную аппроксимацию нелинейной функции.

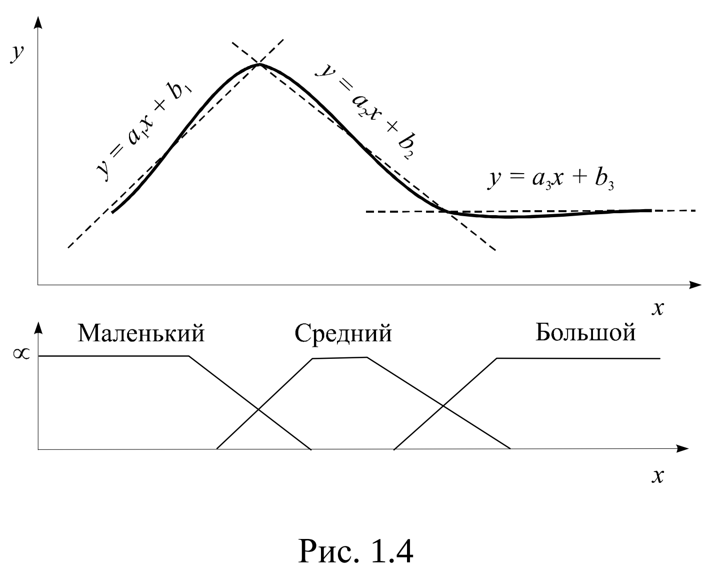

Рис. 1.4

В основе композиционных правил нечеткого логического вывода лежат классические схемы правильных рассуждений *modus ponens* и *modus tollens.*

Классический *modus ponens* – это правило вывода следующего вида:

$$\begin{array}{ll} \textit{предпосылка} & \textbf{если } p, \textbf{ то } q \\ \textit{факт} & p \\ \hline \textit{заключение} & \qquad\qquad q \end{array},$$

которое можно интерпретировать так: если p *истина* и $p \to q$ *истина*, то q также *истинно*. Нечеткая интерпретация всех переменных позволяет перейти к *обобщенному modus ponens:*

$$\begin{array}{ll} \textit{предпосылка} & \textbf{если } x \textit{ есть } A, \textbf{ то } y \textit{ есть } B \\ \textit{факт} & x \textit{ есть } A' \\ \hline \textit{заключение} & \qquad\qquad\qquad y \textit{ есть } B' \end{array},$$

где A, A', B, B' – нечеткие числа, определяемые своими функциями принадлежности на множестве действительных чисел Р.

Заключение B' определяется на основе *операции композиции* в виде

$$B' = A' o \left(A \to B \right)$$

или в терминах функции принадлежности нечетких множеств в виде

$$\forall y \in Y \left(m_{B'}(y) = \sup_{x \in X} \min \left\{ m_{A'}(x), m_{A \to B}(x,y) \right\} \right).$$

Заметим, что вместо $(sup - min)$-композиции можно рассматривать $(sup - T)$-композицию, где T представляет собой *треугольную норму*[3]. В этом случае

$$\forall y \in Y \left(m_{B'}(y) = \sup_{x \in X} T \left\{ m_{A'}(x), m_{A \to B}(x,y) \right\} \right),$$

при этом T не зависит от оператора импликации. На окончательный результат значительное влияние оказывает выбор операций композиции и импликации. Пусть, например, импликация задана в виде импликации Мамдани $x \to y = min(x,y)$, тогда

$$\forall y \in Y \left(m_{B'}(y) = \sup_{x \in X} \min \left\{ m_{A'}(x), m_A(x), m_B(y) \right\} \right) =$$

---

3 *Треугольной нормой (Т-нормой)* называется операция *T: [0,1]´[0,1]®[0,1]*, удовлетворяющая следующим условиям:

a) $T(x,y) = T(y,x)$ – коммутативность;

b) $T((x,y),z) = T(x,(y,z))$ – ассоциативность;

c) $T(0,0) = 0, T(x,1) = T(1,x) = x$ – ограниченность;

d) $(x \le t) \wedge (y \le z) \Rightarrow T(x,y) \le T(t,z)$ – монотонность.

Треугольные нормы используются для моделирования объединения нечетких множеств или дизъюнкции. Двойственным понятием к Т-норме является треугольная *S-конорма*, которая моделирует пересечение или конъюнкцию нечетких множеств. Т-конорма также является коммутативной, ассоциативной, монотонной, но ограниченность задается в виде

$$S(1,1) = 1, S(x,0) = S(0,x) = x.$$

$$\forall y \in Y\left(m_{B'}(y) = min\left\{ m_B(y), \sup_{x\in X}\left(m_A(x), m_{A'}(x)\right)\right\}\right).$$

Пусть база знаний содержит совокупность правил, тогда

$$\begin{array}{l} R_1: \quad \textbf{\textit{если}}\ x\ есть\ A_1, \textbf{\textit{то}}\ y\ есть\ B_1 \\ R_2: \quad \textbf{\textit{если}}\ x\ есть\ A_2, \textbf{\textit{то}}\ y\ есть\ B_2 \\ \dots \\ R_N: \quad \textbf{\textit{если}}\ x\ есть\ A_n, \textbf{\textit{то}}\ y\ есть\ B_N \\ \hfill x\ есть\ A \\ \hline \hfill y\ есть\ B \end{array}.$$

Каждому правилу R_i соответствует импликация $A_i \to B_i$. Существует два подхода к формированию функции принадлежности заключения $m_{B'}(y)$:

Схема 1. Вначале осуществляется агрегирование всех правил с помощью подходящего оператора агрегирования Agg, в результате чего получается некоторое обобщенное правило

$$R = Agg(R_1, R_2, \dots, R_n),$$

а затем применяется оператор композиции

$$B' = A' o\, R = A' o\, Agg(R_1, R_2, \dots, R_n).$$

Схема 2. Второй подход предполагает, что вначале формируется заключение для каждого правила вывода, т.е.

$$\forall i = \overline{1,N}\left(B_i' = A' o\, R_i \right),$$

а затем к полученным компонентам B_i' применяется оператор агрегирования.

Следует отметить, что в настоящее время существует значительный арсенал *операторов агрегирования*[4]. Выбор оператора в первую очередь основывается на той стратегии агрегирования, которая постулируется лицом, принимающим решение, как наиболее подходящая.

На рис. 1.5 представлен *метод нечеткого логического вывода Мамдани* для трех правил, выполненный по схеме 2.

Если результирующее множество B' является нечетким (как в примере), то возникает проблема определения конкретного числового значения выходной переменной. Для этого используется процедура *дефазификации*, которая представляет собой процесс перехода от лингвистического значения переменной к числовому значению на основе использования одного из следующих методов:

– *метод центра тяжести* (*Centre of Gravity – COG*)

---

[4] Под *оператором агрегирования* будем понимать оператор, который векторной оценке ставит в соответствие скалярную величину. Различают конъюнктивную, дизъюнктивную и компромиссную стратегии агрегирования. Каждой стратегии соответствует свой оператор агрегирования. Важнейший класс операторов агрегирования составляют операторы осреднения. Дизъюнктивная стратегия моделируется T-нормами, конъюнктивная – S-конормами.

$$y = \frac{\int\limits_{min}^{max} x \cdot m(x)dx}{\int\limits_{min}^{max} m(x)dx}$$

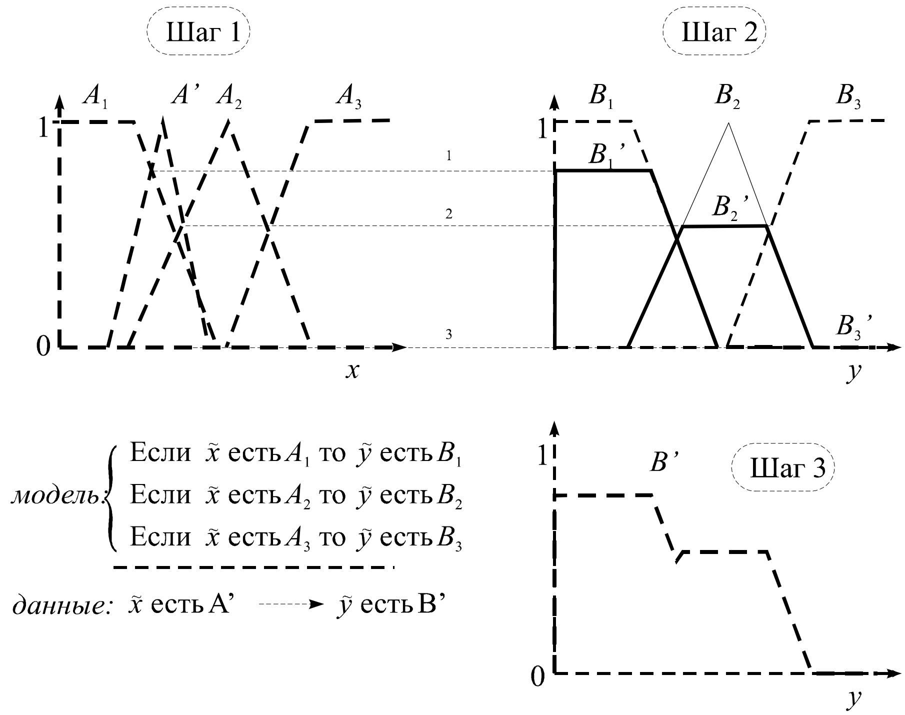

Рис. 1.5

или в случае дискретного представления множеств

$$y = \frac{\sum\limits_{i=1}^{n} x_i \cdot m(x_i)}{\sum\limits_{i=1}^{n} m(x_i)};$$

– *метод центра площади (Centre of Area – COA)*

$$y = \left\{ u : \int\limits_{min}^{u} m(x)dx = \int\limits_{u}^{max} m(x)dx \right\};$$

– *метод левого модального значения (Left Most Maximum – LM)*

$$y = min\{x_m\};$$

– *метод правого модального значения (Right Most Maximum – RM)*

$$y = max\{x_m\};$$

где x_m - модальное значение нечеткого множества.

Замечание: в случае унимодального нечеткого множества левое и правое модальные значения совпадают.

Взаимосвязь всех рассмотренных компонентов нечеткой модели показана на примере нечеткой системы управления (рис. 1.6).

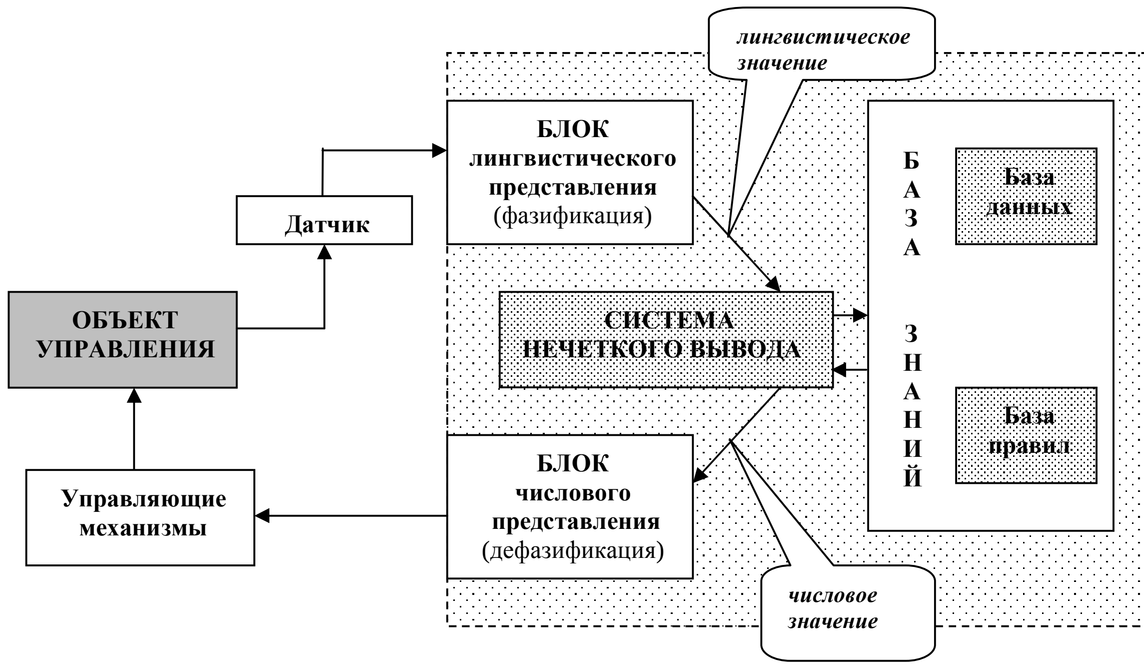

Рис. 1.6

2. РЕШЕНИЕ НЕЧЕТКИХ РЕЛЯЦИОННЫХ УРАВНЕНИЙ

Пусть $X \times Y$ – прямое произведение универсальных множеств. *Нечетким отношением* между множествами X и Y называется функция $R : X \times Y \to [0,1]$, которая ставит в соответствие каждой паре элементов $(x,y) \in X{\times}Y$ величину $m_R(x,y) \in [0,1]$, определяющую степень выполнения отношения xRy. Если X и Y – конечные множества, то нечеткое отношение удобно задавать матрицей $R = \{r_{ij}\}_{n\times m}$, где $n = |X|, m = |Y|$ и $r_{ij} = m_R(x_i, y_j)$.

Введем понятие композиции для нечетких отношений. Пусть $X = \{x_i\}_{i=\overline{1,n}}$, $Y = \{y_j\}_{j=\overline{1,m}}$, $Z = \{z_l\}_{l=\overline{1,k}}$ – конечные множества; A, R и B – нечеткие отношения, определенные на $X{\times}Y$, $Y{\times}Z$ и $X{\times}Z$ соответственно. Пусть заданы матричные представления данных отношений

$$A = \{a_{ij} \mid a_{ij}: (x_i, y_j) \to [0,1]\},$$

$$R = \{r_{jl} \mid r_{jl}: (y_j, z_l) \to [0,1]\},$$

$$B = \{b_{il} \mid b_{il}: (x_i, z_l) \to [0,1]\}.$$

Max-T *композицией* нечетких отношений A и R называется нечеткое отношение $A \circ R$ на $X{\times}Z$, определяемое в виде

$$(A \circ R)(x,z) = \max_{y\in Y}\{T(A(x,y), R(y,z))\},$$

где T – треугольная T-норма.

Выбор T-нормы обусловливает тип композиции. Наиболее распространенными являются *максминная композиция*

$$\max_{y \in Y} \min\{A(x,y), R(y,z)\}$$

при $T(x,y) = \min\{x,y\}$,

максмультипликативная композиция

$$\max_{y \in Y}\left(A(x,y) \cdot R(y,z)\right)$$

при $T(x,y) = x \cdot y$.

Пусть A, R и B – нечеткие отношения. Рассмотрим выражение

$$A \circ R = B,$$

которое также можно записать в виде

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix} \circ \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1k} \\ r_{21} & r_{22} & \dots & r_{2k} \\ \dots & \dots & \dots & \dots \\ r_{m1} & r_{m2} & \dots & r_{mk} \end{bmatrix} = \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1k} \\ b_{21} & b_{22} & \dots & b_{2n} \\ \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & b_{nk} \end{bmatrix}.$$

Если известны A, B и задан тип композиции $\circ$, а R подлежит определению, то это выражение называют *нечетким реляционным уравнением*. В матричном представлении оно запишется так

$$\max_{j}(T(a_{ij}, r_{jl})) = b_{il} \quad \left(i = \overline{1,n}, j = \overline{1,m}, l = \overline{1,k}\right).$$

Аппарат нечетких реляционных уравнений используется для моделирования технических, социальных, экономических, экологических и других процессов. К уравнениям подобного типа сводятся многие прикладные задачи, например, распознавание образов, сжатие изображений, отдельные задачи компьютерной поддержки постановки медицинского диагноза. Возможность использования в качестве исходных данных информации, полученной от экспертов, позволила широко использовать нечеткие уравнения в задачах принятия решений и задачах диагностического характера.

Пусть $R(A,B) = \{R \mid A \circ R = B\}$ – *множество решений* нечеткого реляционного уравнения. Известно, что если $R(A,B)$ не пусто, то оно имеет единственное максимальное решение $\hat{R}$ и конечное число минимальных решений $\check{R}(A,B)$, через которые все остальные решения определяются следующим образом:

$$R(A,B) = \bigcup_{\check{R} \in \check{R}(A,B)} \{R \in R(A,B) \mid \check{R} \le R \le \hat{R}\}.$$

Максимальным решением нечеткого реляционного уравнения является максимальный[5] элемент множества решений этого уравнения, а *минимальными*

---

[5] Элемент x подмножества X' частично упорядоченного множества X называется *максимальным* элементом X', если для каждого элемента y Î X из того что y ³ x следует, что $y=x$. Другими словами, во множестве Y нет элемента, который был бы «больше» x.

решениями – минимальные[6] элементы множества решений $R(A,B)$. Таким образом, решить нечеткое реляционное уравнение означает найти его максимальное и все минимальные решения.

Нечеткое реляционное уравнение $A \circ R = B$ обычно представляют в виде системы более простых независимых уравнений:

$$A \circ \boldsymbol{r}_l = \boldsymbol{b}_l \quad \left(l = \overline{1,k}\right),$$

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix} \circ \begin{bmatrix} \dots\, r_{1l}\, \dots \\ \dots\, r_{2l}\, \dots \\ \dots\dots\dots \\ \dots\, r_{ml}\, \dots \end{bmatrix} = \begin{bmatrix} \dots\, b_{1l}\, \dots \\ \dots\, b_{2l}\, \dots \\ \dots\dots\dots \\ \dots\, b_{ml}\, \dots \end{bmatrix},$$

где $\boldsymbol{r}_l$ и $\boldsymbol{b}_l$ – l-ые столбцы матриц R и B соответственно.

Максимальное решение уравнения $A \circ R = B$ определяется в виде $\hat{R} = [\hat{\boldsymbol{r}}_1 \hat{\boldsymbol{r}}_2 \dots \hat{\boldsymbol{r}}_k]$, где $\hat{r}_l$ $\left(l = \overline{1,k}\right)$ – максимальное решение l-го уравнения системы. Множество минимальных решений определяется как множество матриц вида $[\check{\boldsymbol{r}}_1 \check{\boldsymbol{r}}_2 \dots \check{\boldsymbol{r}}_k]$, где $\check{r}_l \in \check{r}(A, b_l)$ – одно из минимальных решений l-го уравнения системы. Рассмотрим один из методов решения нечетких реляционных уравнений, который относится к численным методам. Он используется при решении уравнений с максминной или максмультипликативной композициями. Представим исходное уравнение $A \circ R = B$ в виде системы независимых уравнений

$$A \circ \boldsymbol{r}_l = \boldsymbol{b}_l \quad \left(l = \overline{1,k}\right),$$

где $\boldsymbol{r}_l$ и $\boldsymbol{b}_l$ – l-ые столбцы матриц R и B соответственно.

Метод позволяет найти максимальное и множество минимальных решений нечетких реляционных уравнений вида $A \circ \boldsymbol{r} = \boldsymbol{b}$. В основе метода лежит следующая <u>теорема</u>: максимальное решение уравнения $A \circ \boldsymbol{r} = \boldsymbol{b}$ $\hat{r} = (\hat{r}_1, .., \hat{r}_m)$ находится в виде

$$\hat{r}_j = \min_{i = \overline{1,n}} \sigma(a_{ij}, b_i) \quad \left(j = \overline{1,m}\right),$$

где $\sigma(a_{ij}, b_i) = \begin{cases} b_i, & \text{если } a_{ij} > b_i \\ 1, & \text{иначе} \end{cases}$ или $\sigma(a_{ij}, b_i) = \begin{cases} \dfrac{b_i}{a_{ij}}, & \text{если } a_{ij} > b_i \\ 1, & \text{иначе.} \end{cases}$ (первая формула используется для максминной композиции, вторая – для максмультипликативной композиции)

После нахождения максимального решения для отыскания множества минимальных решений используется <u>метод «матричного шаблона»</u>:

---

[6] Элемент x подмножества X' частично упорядоченного множества X называется *минимальным* элементом X', если для каждого элемента $y \hat{I} X$ из того что $y \pounds x$ следует, что $y=x$. Другими словами, во множестве Y нет элемента, который был бы «меньше» x.

Обычно термины *минимальный* и *максимальный* элемент относятся ко всему множеству, т.е. $X'=X$.

Шаг 1. Упорядочить элементы столбца $\boldsymbol{b}$ и строки матрицы A по убыванию элементов столбца и найти наибольшее решение $\hat{\boldsymbol{r}}$.

Шаг 2. Составить «матричный шаблон» – матрицу M = [m_{ij}] ($i=\overline{1,n}$, $j=\overline{1,m}$), где m_{ij} представляет собой пару вида $m_{ij}=(a_{ij},\hat{r}_j)$:

$$M=\begin{pmatrix} (a_{11},\hat{r}_1) & \cdots & (a_{1m},\hat{r}_m) \\ \vdots & \ddots & \vdots \\ (a_{n1},\hat{r}_1) & \cdots & (a_{nm},\hat{r}_m,) \end{pmatrix}.$$

Шаг 3. Пометить m_{ij} такие, что $\min(a_{ij},\hat{r}_j)=b_i$ (или $a_{ij}\cdot\hat{r}_j=b_i$ для максмультипликативной композиции). Обозначим помеченные элементы матричного шаблона $\tilde{m}_{ij}$.

Шаг 4. Если i_1 – наименьший среди всех помеченных элементов $\tilde{m}_{ij}$, тогда в качестве $\check{r}_{j_1}$ возьмем наименьший элемент в паре $\tilde{m}_{i_1j_1}$ (для максмультипликативной композиции $\check{r}_{j_1}=\hat{r}_{j_1}$).

Шаг 5. Вычеркнуть i_1-ю строку и j_1-й столбец матрицы M, затем вычеркнуть все строки, включающие в себя помеченные $\tilde{m}_{ij_1}$, где $i\neq i_1$.

Шаг 6. Во всех оставшихся и помеченных $\tilde{m}_{ij}$ найти наименьший i и обозначить его i_2, тогда в качестве $\check{r}_{j_2}$ возьмем наименьший элемент в паре $\tilde{m}_{i_2j_2}$ (для максмультипликативной композиции $\check{r}_{j_2}=\hat{r}_{j_2}$).

Шаг 7. Вычеркнуть i_2-ю строку и j_2-й столбец матрицы M, затем вычеркнуть все строки, включающие в себя помеченные $\tilde{m}_{ij_2}$, где $i\neq i_2$.

Шаг 8. Повторять шаги 6 и 7 до тех пор, пока в матричном шаблоне есть помеченные, но не вычеркнутые $\tilde{m}_{ij}$.

Шаг 9. Все $\check{r}_j$, не получившие значение в шагах 4–7, изменить на 0.

Шаг 10. Если строка i_1 (или i_2) содержит более одного помеченного $\tilde{m}_{i_1j}$ ($\tilde{m}_{i_2j}$), выбрать другой помеченный $\tilde{m}_{i_1j}$ ($\tilde{m}_{i_2j}$) и повторить шаги 4–9.

Шаг 11. Повторять шаги 1–10, пока все минимальные решения не будут получены.

Пример. Решим уравнение

$$\begin{pmatrix} 0{,}9 & 0{,}5 & 0{,}4 & 0{,}4 & 0{,}9 \\ 0{,}2 & 0{,}3 & 0 & 0{,}6 & 0{,}2 \\ 0{,}6 & 0{,}7 & 1 & 0{,}7 & 0{,}1 \end{pmatrix} \circ r = \begin{pmatrix} 0{,}72 \\ 0{,}3 \\ 0{,}9 \end{pmatrix},$$

где $\circ$ – максминная композиция, $A=\begin{pmatrix} 0{,}9 & 0{,}5 & 0{,}4 & 0{,}4 & 0{,}9 \\ 0{,}2 & 0{,}3 & 0 & 0{,}6 & 0{,}2 \\ 0{,}6 & 0{,}7 & 1 & 0{,}7 & 0{,}1 \end{pmatrix}$, $b=\begin{pmatrix} 0{,}72 \\ 0{,}3 \\ 0{,}9 \end{pmatrix}$.

Шаг 1.

$$b=\begin{pmatrix}0{,}9\\0{,}72\\0{,}3\end{pmatrix},\ A=\begin{pmatrix}0{,}6 & 0{,}7 & 1 & 0{,}7 & 0{,}1\\0{,}9 & 0{,}5 & 0{,}4 & 0{,}4 & 0{,}9\\0{,}2 & 0{,}3 & 0 & 0{,}6 & 0{,}2\end{pmatrix}.$$

Найдем $s=\begin{pmatrix}1 & 1 & 0{,}9 & 1 & 1\\0{,}72 & 1 & 1 & 1 & 0{,}72\\1 & 1 & 1 & 0{,}3 & 1\end{pmatrix}$, получим

$$\hat{r}=\begin{pmatrix}0{,}72\\1\\0{,}9\\0{,}3\\0{,}72\end{pmatrix}$$ – максимальное решение.

Шаг 2.

$$M=\begin{pmatrix}0{,}6\,|\,0{,}72 & 0{,}7\,|\,1 & 1\,|\,0{,}9 & 0{,}7\,|\,0{,}3 & 0{,}1\,|\,0{,}72\\0{,}9\,|\,0{,}72 & 0{,}5\,|\,1 & 0{,}4\,|\,0{,}9 & 0{,}4\,|\,0{,}3 & 0{,}9\,|\,0{,}72\\0{,}2\,|\,0{,}72 & 0{,}3\,|\,1 & 0\,|\,0{,}9 & 0{,}6\,|\,0{,}3 & 0{,}2\,|\,0{,}72\end{pmatrix}.$$

Шаг 3. Помечаем элементы:

$$M=\begin{pmatrix}0{,}6\,|\,0{,}72 & 0{,}7\,|\,1 & \underline{1\,|\,0{,}9} & 0{,}7\,|\,0{,}3 & 0{,}1\,|\,0{,}72\\\underline{0{,}9\,|\,0{,}72} & 0{,}5\,|\,1 & 0{,}4\,|\,0{,}9 & 0{,}4\,|\,0{,}3 & \underline{0{,}9\,|\,0{,}72}\\0{,}2\,|\,0{,}72 & \underline{0{,}3\,|\,1} & 0\,|\,0{,}9 & \underline{0{,}6\,|\,0{,}3} & 0{,}2\,|\,0{,}72\end{pmatrix}.$$

Шаг 4-1. Первый помеченный элемент находится в первой строке и третьем столбце. Согласно алгоритму $r_3=0.9$.

Шаг 5-1. Вычеркиваем строки и столбцы согласно алгоритму. Получим следующую матрицу:

$$M=\begin{pmatrix}0{,}6\,|\,0{,}72 & 0{,}7\,|\,1 & \underline{1\,|\,0{,}9} & 0{,}7\,|\,0{,}3 & 0{,}1\,|\,0{,}72\\\underline{0{,}9\,|\,0{,}72} & 0{,}5\,|\,1 & 0{,}4\,|\,0{,}9 & 0{,}4\,|\,0{,}3 & \underline{0{,}9\,|\,0{,}72}\\0{,}2\,|\,0{,}72 & \underline{0{,}3\,|\,1} & 0\,|\,0{,}9 & \underline{0{,}6\,|\,0{,}3} & 0{,}2\,|\,0{,}72\end{pmatrix}.$$

Шаг 6-1. Первый помеченный элемент находится во второй строке и первом столбце. Согласно алгоритму $r_1=0{,}72\ \left(r_3=0{,}9\right)$.

Шаг 7-1. Вычеркивая строки и столбцы согласно алгоритму, получим следующую матрицу:

$$M=\begin{pmatrix}0{,}6\,|\,0{,}72 & 0{,}7\,|\,1 & \underline{1\,|\,0{,}9} & 0{,}7\,|\,0{,}3 & 0{,}1\,|\,0{,}72\\\underline{0{,}9\,|\,0{,}72} & 0{,}5\,|\,1 & 0{,}4\,|\,0{,}9 & 0{,}4\,|\,0{,}3 & \underline{0{,}9\,|\,0{,}72}\\0{,}2\,|\,0{,}72 & \underline{0{,}3\,|\,1} & 0\,|\,0{,}9 & \underline{0{,}6\,|\,0{,}3} & 0{,}2\,|\,0{,}72\end{pmatrix}.$$

Шаг 8-1. Переходим к шагу 6.

Шаг 6-2. Первый помеченный элемент находится в третьей строке и втором столбце. Согласно алгоритму $r_2 = 0{,}3 \ \left(r_1 = 0{,}72, r_3 = 0{,}9 \right)$.

Шаг 7-2. Вычеркиваем строки и столбцы согласно алгоритму. Получим следующую матрицу:

$$M = \begin{pmatrix} 0{,}6|0{,}72 & 0{,}7|1 & \underline{1|0{,}9} & 0{,}7|0{,}3 & 0{,}1|0{,}72 \\ \underline{0{,}9|0{,}72} & 0{,}5|1 & 0{,}4|0{,}9 & 0{,}4|0{,}3 & \underline{0{,}9|0{,}72} \\ 0{,}2|0{,}72 & \underline{0{,}3|1} & 0|0{,}9 & \underline{0{,}6|0{,}3} & 0{,}2|0{,}72 \end{pmatrix}.$$

Все вычеркнуто. Значит, получили минимальное решение.

Шаг 9-1.

$$\breve{r}_1 = \begin{pmatrix} 0{,}72 \\ 0{,}3 \\ 0{,}9 \\ 0 \\ 0 \end{pmatrix}.$$

Шаг 10-1. В третьей строке остался помеченный элемент в четвертом столбце.

Шаг 4-2. Помечаем элемент в третьей строке и четвертом столбце. Согласно алгоритму $r_4 = 0{,}3 \ \left(r_1 = 0{,}72, r_3 = 0{,}9 \right)$.

Шаг 5-2.

$$M = \begin{pmatrix} 0{,}6|0{,}72 & 0{,}7|1 & \underline{1|0{,}9} & 0{,}7|0{,}3 & 0{,}1|0{,}72 \\ \underline{0{,}9|0{,}72} & 0{,}5|1 & 0{,}4|0{,}9 & 0{,}4|0{,}3 & \underline{0{,}9|0{,}72} \\ 0{,}2|0{,}72 & \underline{0{,}3|1} & 0|0{,}9 & \underline{0{,}6|0{,}3} & 0{,}2|0{,}72 \end{pmatrix}.$$

Шаг (6-8)-2. Все вычеркнуто. Значит, получили второе минимальное решение.

Шаг 9-2.

$$\breve{r}_2 = \begin{pmatrix} 0{,}72 \\ 0 \\ 0{,}9 \\ 0{,}3 \\ 0 \end{pmatrix}.$$

Шаг 10-2. Во второй строке остался помеченный элемент в пятом столбце.

Шаг 4-3. Помечаем элемент во второй строке и пятом столбце. Согласно алгоритму $r_5 = 0{,}72 \ \left(r_3 = 0{,}9 \right)$.

Шаг 5-3.

$$M = \begin{pmatrix} 0{,}6|0{,}72 & 0{,}7|1 & \underline{1|0{,}9} & 0{,}7|0{,}3 & 0{,}1|0{,}72 \\ \underline{0{,}9|0{,}72} & 0{,}5|1 & 0{,}4|0{,}9 & 0{,}4|0{,}3 & \underline{0{,}9|0{,}72} \\ 0{,}2|0{,}72 & \underline{0{,}3|1} & 0|0{,}9 & \underline{0{,}6|0{,}3} & 0{,}2|0{,}72 \end{pmatrix}$$

Шаг 6-3. Первый помеченный элемент находится в третьей строке и втором столбце. Согласно алгоритму $r_2 = 0{,}3 \ \left(r_5 = 0{,}72, r_3 = 0{,}9 \right)$.

<u>Шаг 7-3.</u> Вычеркиваем строки и столбцы согласно алгоритму. Получим следующую матрицу:

$$M=\begin{pmatrix} 0{,}6|0{,}72 & 0{,}7|1 & \underline{1|0{,}9} & 0{,}7|0{,}3 & 0{,}1|0{,}72 \\ \underline{0{,}9|0{,}72} & 0{,}5|1 & 0{,}4|0{,}9 & 0{,}4|0{,}3 & \underline{0{,}9|0{,}72} \\ 0{,}2|0{,}72 & \underline{0{,}3|1} & 0|0{,}9 & \underline{0{,}6|0{,}3} & 0{,}2|0{,}72 \end{pmatrix}.$$

<u>Шаг 8-3.</u> Все вычеркнуто. Значит, получили третье минимальное решение.

<u>Шаг 9-3.</u>

$$\overset{(}{r}_3=\begin{pmatrix} 0 \\ 0{,}3 \\ 0{,}9 \\ 0 \\ 0{,}72 \end{pmatrix}.$$

<u>Шаг 10-3.</u> В третьей строке остался помеченный элемент в четвертом столбце.

<u>Шаг 4-4.</u> Помечаем элемент в третьей строке и четвертом столбце. Согласно алгоритму $r_4=0{,}3\ \left(r_5=0{,}72, r_3=0{,}9\right)$.

<u>Шаг 5-4.</u>

$$M=\begin{pmatrix} 0{,}6|0{,}72 & 0{,}7|1 & \underline{1|0{,}9} & 0{,}7|0{,}3 & 0{,}1|0{,}72 \\ \underline{0{,}9|0{,}72} & 0{,}5|1 & 0{,}4|0{,}9 & 0{,}4|0{,}3 & \underline{0{,}9|0{,}72} \\ 0{,}2|0{,}72 & \underline{0{,}3|1} & 0|0{,}9 & \underline{0{,}6|0{,}3} & 0{,}2|0{,}72 \end{pmatrix}.$$

<u>Шаг (6-8)-4.</u> Все вычеркнуто. Получили четвертое минимальное решение.

<u>Шаг 9-4.</u>

$$\overset{(}{r}_4=\begin{pmatrix} 0 \\ 0 \\ 0{,}9 \\ 0{,}3 \\ 0{,}72 \end{pmatrix}.$$

<u>Шаг 10-4, Шаг 11.</u> Все вычеркнуто. Больше помеченных элементов нет. Получили четыре минимальных решения.

<u>Ответ:</u>

$$\overset{)}{r}=\begin{pmatrix} 0{,}72 \\ 1 \\ 0{,}9 \\ 0{,}3 \\ 0{,}72 \end{pmatrix},\quad \overset{(}{r}=\begin{pmatrix} 0{,}72 & 0{,}72 & 0 & 0 \\ 0{,}3 & 0 & 0{,}3 & 0 \\ 0{,}9 & 0{,}9 & 0{,}9 & 0{,}9 \\ 0 & 0{,}3 & 0 & 0{,}3 \\ 0 & 0 & 0{,}72 & 0{,}72 \end{pmatrix}.$$

3. НЕЧЕТКАЯ КЛАСТЕРИЗАЦИЯ

Кластерный анализ предназначен для разбиения множества объектов на заданное или неизвестное число классов на основе некоторого критерия качества классификации, который отражает следующие требования:

– внутри классов объекты должны быть тесно связаны между собой;
– объекты разных классов должны быть далеки друг от друга;
– распределение объектов по классам должно быть более или менее равномерным.

Характерные представители классов называются *прототипами*. В метрическом пространстве «близость» обычно определяют через функцию расстояние, которое рассчитывается как между объектами заданного множества, так и от этих объектов к прототипу класса. Как правило, координаты прототипов заранее неизвестны – они находятся одновременно с разбиением данных на классы. Существует множество методов классификации, которые можно разбить на четкие и нечеткие. Четкие методы разбивают исходное множество объектов на несколько непересекающихся подмножеств, при этом любой объект из исходного множества принадлежит только одному классу. Нечеткие методы позволяют одному и тому же объекту принадлежать одновременно нескольким (или даже всем) классам, но с различной степенью.

Пусть $A=\{a_1,a_2,...,a_n\}$ – заданное множество объектов. Исходной информацией для кластеризации является *матрица данных*, в которой каждая строка соответствует объекту и содержит оценки по каждому из m признаков, всесторонне характеризующих объекты из множества A

$$X=\begin{bmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & & & \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{bmatrix}.$$

Через x_k будем обозначать векторную оценку объекта a_k. Задача нечеткого кластерного анализа формулируется следующим образом: на основе исходных данных необходимо определить такое *нечеткое разбиение* $\Re(A)=\{A_k\}$ или *нечеткое покрытие* $\Im(A)=\{a_k\}$ множества А, которое доставляет экстремум некоторой функции, определяющей качество классификации. Нечеткие кластеры A_k из $\Re(A)$ или $\Im(A)$ определяются с помощью функций принадлежности $\mu_{A_k}: A\to[0,1]$.

Приведенная постановка является достаточно общей, поэтому ограничимся конкретными предположениями.

а) Пусть количество кластеров c заранее известно. Будем считать, что нечеткие кластеры образуют нечеткое покрытие множества A, формализуя это понятие с помощью следующего условия:

$$\forall a_i\left(\sum_{k=1}^{c} m_{A_K}(a_i)=1\right).$$

б) Введем дополнительные ограничения на функции принадлежности

$$\forall k \in \overline{2,c} \ \left(\sum_{i=1}^{n} m_{A_K}(a_i) > 0 \right),$$

$$\forall a_i \in A \ \forall k \in \overline{2,c} \ \left(m_{A_K}(a_i) \geq 0 \right).$$

Первое условие исключает появление пустых нечетких кластеров, второе имеет чисто формальный характер и вытекает из естественных требований к функции принадлежности нечеткого множества.

в) Пусть *критерий качества классификации* представляется в виде суммы квадратов взвешенных отклонений координат объектов кластеризации от центров искомых нечетких кластеров

$$F = \sum_{k=1}^{n} \sum_{i=1}^{c} (m_{ik})^m \left\| x_k - n_i \right\|^2 ,$$

при этом центры классов рассчитываются по формуле

$$n_i = \frac{1}{\sum_{k=1}^{n} (m_{ik})^m} \sum_{k=1}^{n} (m_{ik})^m x_k \quad \left(i = \overline{1,c} \right).$$

Величина m носит название экспоненциального веса и задается пользователем. Она уменьшает влияние «шума» при вычислении центров кластеров и значения целевой функции: большие значения степеней принадлежности увеличиваются, а маленькие – уменьшаются. Следует задавать $m > 1$. Чем больше m, тем менее нечеткую классификацию мы получаем.

Теперь задача нечеткой кластеризации может быть сформулирована следующим образом: на основе исходных данных (матрицы X, количества нечетких кластеров *c* (*c*>*1*), параметра m), определить матрицу D значений функций принадлежности объектов кластеризации $a_i \in A$ нечетким кластерам A_k, которые минимизируют функцию *F*.

Для ее решения в пакете *MatLab* реализован алгоритм **Fuzzy C-means (FCM-алгоритм)**, вычислительная схема которого приводится ниже.

Шаг 1. Установить параметры алгоритма: c – количество классов; m – экспоненциальный вес; e – параметр сходимости алгоритма.

Ш*аг 2.* Случайным образом сгенерировать матрицу нечеткого разбиения *U*, удовлетворяющую условиям пункта б).

Шаг 3. Рассчитать центры кластеров $\{v_i\} \quad \left(i = \overline{1,c} \right)$.

Шаг 4. Рассчитать расстояния между объектами из *A* и центрами кластеров:

$$D_{ki} = \sqrt{\left\| x_k - v_i \right\|^2} \quad \left(k = \overline{1,m},\ i = \overline{1,c} \right).$$

Шаг 5. Пересчитать элементы матрицы нечеткого разбиения $U \left(i = \overline{1,c}, k = \overline{1,m} \right)$:

если $D_{ki} > 0 : m_{ki} = \dfrac{1}{\left(D_{ik}^2 \cdot \sum\limits_{j=\overline{1,c}} \dfrac{1}{D_{jk}^2} \right)^{1/(m-1)}}$;

если $D_{ki} = 0 : m_{kj} = \begin{cases} 1, j = i \\ 0, j \neq i \end{cases} \quad \left(j = \overline{1,c}\right).$

Шаг 6. Проверить: если $\left\| U^{l+1} - U^{l} \right\| \leq \varepsilon$ (здесь U^{l+1}, U^{l} - матрицы нечеткого разбиения, построенные на l и $(l+1)$ итерациях соответственно), то нечеткая классификация получена, иначе перейти к шагу 3.

Другой алгоритм **Subtractive clustering** используется в том случае, если заранее количество нечетких кластеров определить нельзя. Его идея состоит в том, что каждая точка данных предполагается в качестве центра потенциального кластера, после этого вычисляется количественная мера способности каждой точки данных представлять центр кластера, которая основана на оценке плотности точек данных вокруг данной точки.

Вычислительная схема основана на выполнении следующих действий.

Шаг 1. Для каждой точки определить потенциал становления ее центром кластера (точка с большим количеством соседей будет иметь высокий потенциал)

$$P_i = \sum_{j=1}^{n} e^{-a\left\| x_i - x_j \right\|^2},$$

где x_i – точка данных, $a = \frac{4}{r_a^{\ 2}}$, $\|\cdot\|$– евклидово расстояние, r_a – положительная константа, задающая радиус (точки вне этого радиуса оказывают незначительно влияние на потенциал).

Шаг 2. Выбрать точку данных с максимальным потенциалом для определения центра очередного кластера.

Шаг 3. Чтобы определить следующий кластер и координаты его центра, изменить потенциал каждой точки следующим образом:

$$P_i^{new} = P_i - P_k^* e^{-\beta\left\| x_i - x_k^* \right\|^2},$$

где P_k^* - потенциал найденного на предыдущем шаге центра кластера; x_k^* - координаты центра этого кластера, $b = \frac{4}{r_b^2}$, r_b - положительная константа.

Замечание: теперь точки, расположенные близко к центру найденного кластера, имеют сильно уменьшенный потенциал, и поэтому не будут выбраны в качестве центра следующего кластера. Чтобы избежать близко расположенных кластеров выбирают $r_b = 1.25r$.

Шаг 4. Если максимальный потенциал не меньше некоторого заданного уровня от первого потенциала (например, $P_k^* \leq 0.15P_1^*$), то точка с максимальным потенциалом объявляется центром следующего кластера, и возвращаются к шагу 3. Иначе все возможные кластеры определены.

4. ЛАБОРАТОРНЫЙ ПРАКТИКУМ

4.1. Лабораторная работа №1

Проектирование нечетких систем Мамдани

В среде *MatLab* присутствуют следующие пять основных средств графического интерфейса пользователя, которые обеспечивают процесс нечеткого моделирования.

Редактор системы нечеткого вывода предполагает ввод и редактирование количества входных и выходных переменных и наименований переменных.

Редактор функции принадлежности используется для определения формы функции принадлежности, ассоциированной с каждым термом лингвистической переменной.

Редактор правил вывода применяется для редактирования списка правил, описывающих поведение моделируемой системы.

Средство просмотра правил вывода используется в целях диагностики и может показывать, например, активность правил или форму влияния отдельных функций принадлежности на результат нечеткого вывода.

Средство просмотра поверхности вывода используется для отображения зависимости выходной переменной от одной или двух входных переменных.

Эти средства связаны между собой динамически, и производимые изменения в одном из них влекут изменения в других.

Рассмотрим основные этапы проектирования нечетких систем Мамдани на примере создания системы нечеткого логического вывода, моделирующей зависимость

$$y = x_1^2 \cdot sin(x_2 - 1),\ x_1 \in [-7,3],\ x_2 \in [-4.4,1.7],$$

для представления трехмерного изображения которой используем следующую программу:

```
% Построение графика функции y=x1^2*sin(x2-1)
% в области x1є[-7,3] и x2є[-4.4,1.7].
n=15;
x1=-7:10/(n-1):3;
x2=-4.4:6.1/(n-1):1.7;
y=zeros(n,n);
for j=1:n
y(j,:)=x1.^2*sin(x2(j)-1);
end
surf(x1,x2,y)
xlabel('x1')
ylabel('x2')
zlabel('y')
title('Target');
```

В результате выполнения программы получим графическое изображение, приведенное на рис. 4.1.1.

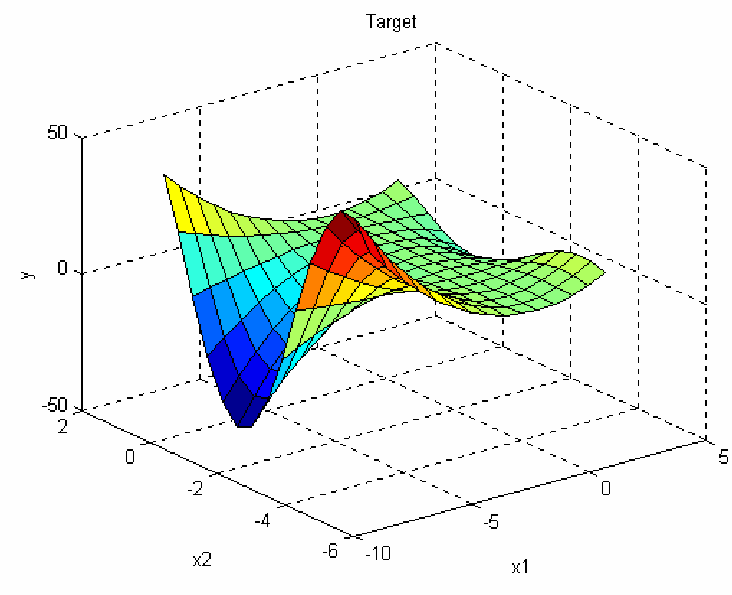

Рис. 4.1.1

Проектирование системы нечеткого логического вывода, соответствующей приведенному графику, состоит в выполнении следующей последовательности шагов.

Шаг 1. Для загрузки основного fis-редактора введем команду **fuzzy** в командной строке. После этого откроется новое графическое окно, показанное на рис. 4.1.2.

Шаг 2. Добавим вторую входную переменную. Для этого в меню **Edit** выбираем команду **Add input**.

Шаг 3. Переименуем первую входную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке **input1**, введем новое обозначение **x1** в поле редактирования имени текущей переменной и нажмем **<Enter>**.

Шаг 4. Переименуем вторую входную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке **input2**, введем новое обозначение **x2** в поле редактирования имени текущей переменной и нажмем **<Enter>**.

Шаг 5. Переименуем выходную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке **output1**, введем новое обозначение **y** в поле редактирования имени текущей переменной и нажмем **<Enter>**.

Шаг 6. Зададим имя системы. Для этого в меню **File** выбираем в подменю **Export** команду **To disk** и вводим имя файла, например, **first**.

Шаг 7. Перейдем в редактор функций принадлежности. Для этого сделаем двойной щелчок левой кнопкой мыши на блоке **x1**.

Шаг 8. Зададим диапазон изменения переменной **x1**. Для этого введем **-7 3** в поле **Range** (рис. 4.1.3) и нажмем **<Enter>**.

Шаг 9. Зададим функции принадлежности переменной **x1**. Для лингвистической оценки этой переменной будем использовать 3 терма с треугольными функциями принадлежности. Для этого в меню **Edit** выберем команду **Add MFs...** В результате появится диалоговое окно выбора типа и количества

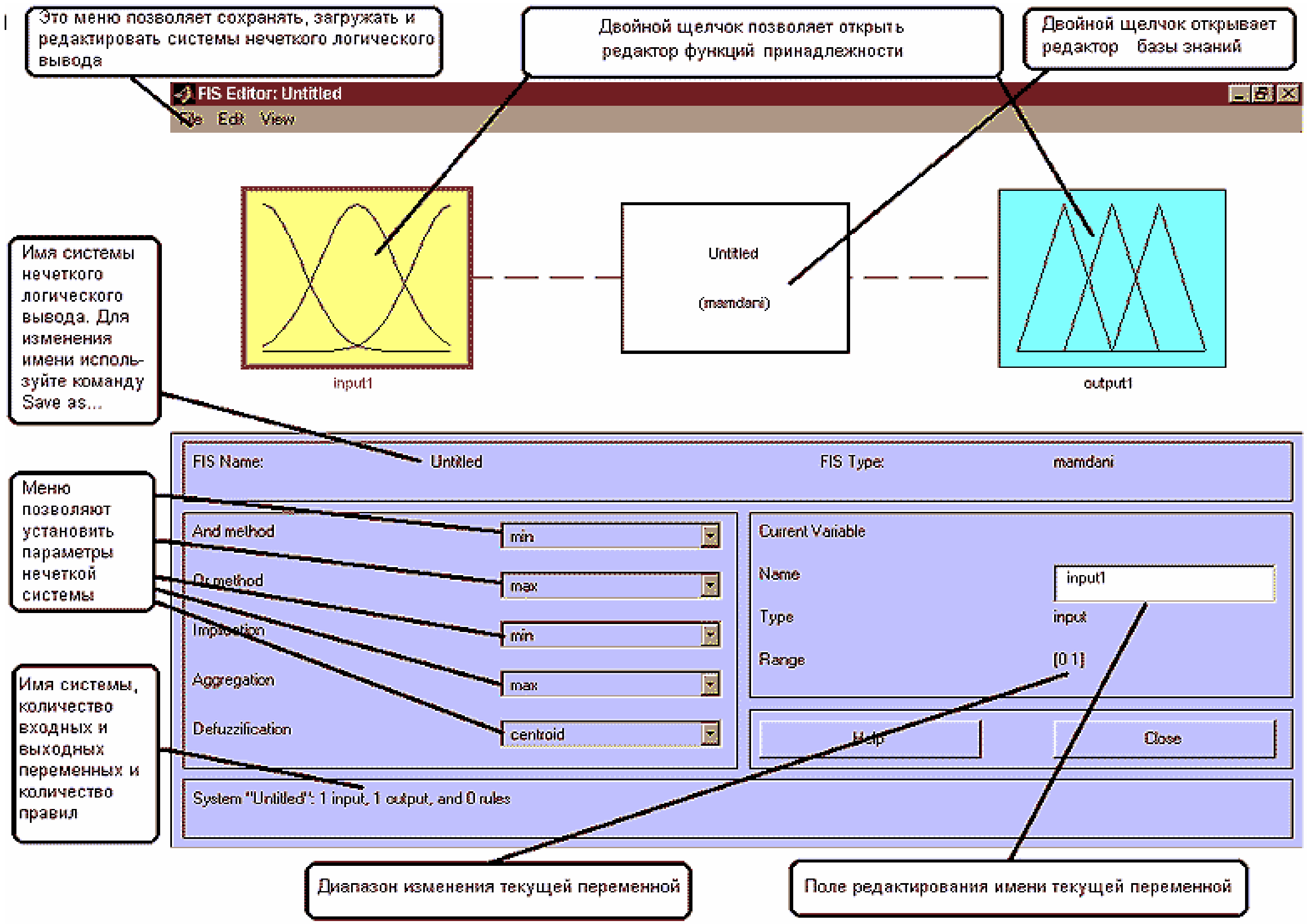

Рис. 4.1.2. Окно редактора FIS-Editor

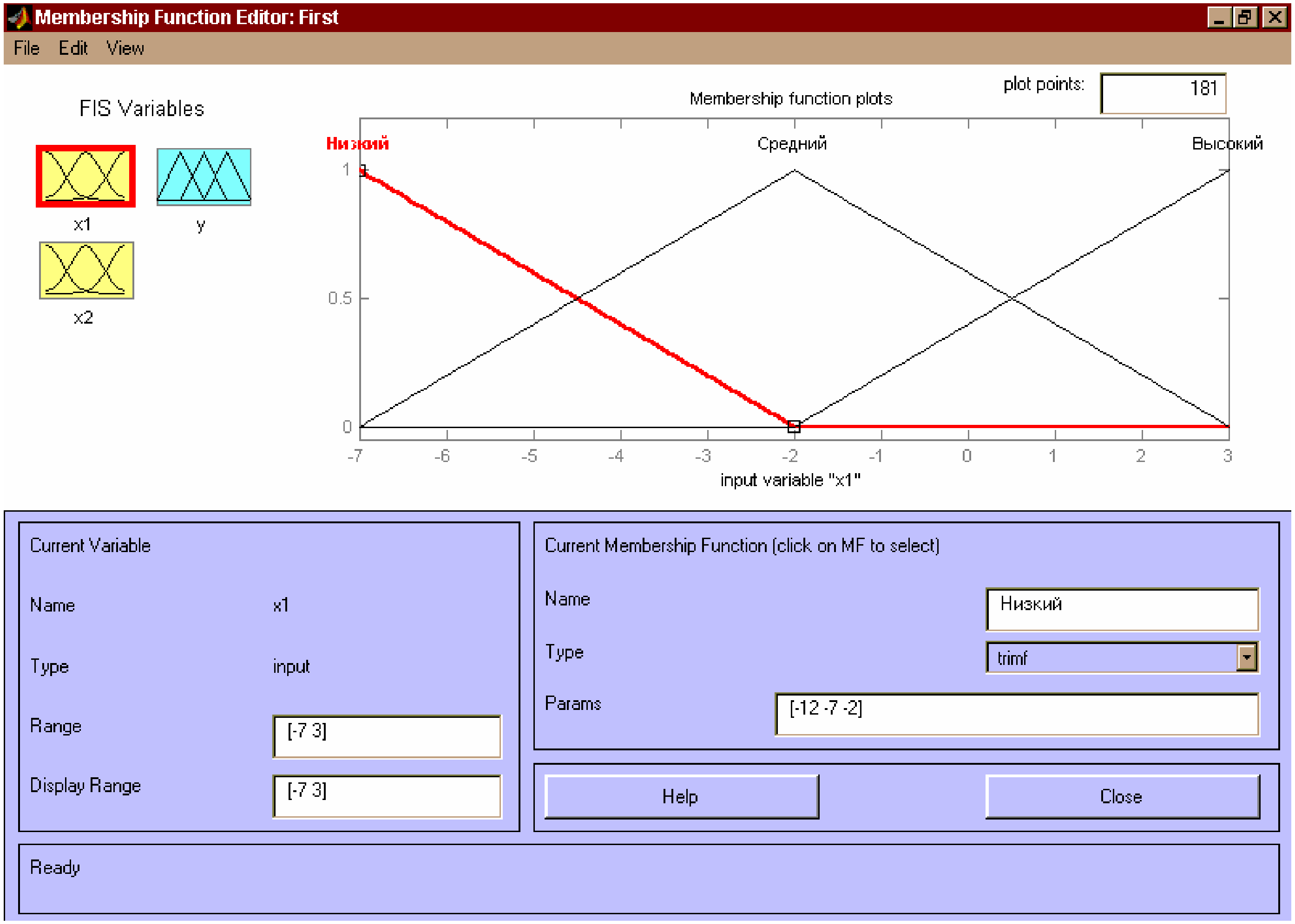

Рис. 4.1.3. Функции принадлежности переменной x1

функций принадлежности. По умолчанию это 3 терма с треугольными функциями принадлежности. Поэтому просто нажимаем **<Enter>.**

Шаг 10. Зададим наименования термов переменной **x1**. Для этого делаем один щелчок левой кнопкой мыши по графику первой функции принадлежности (рис. 4.1.3). Затем вводим наименование терма, например, *низкий* в поле **Name** и нажмем **<Enter>**. После делаем один щелчок левой кнопкой мыши по графику второй функции принадлежности и вводим наименование терма, например, *средний* в поле **Name** и нажмем **<Enter>**. Еще раз делаем один щелчок левой кнопкой мыши по графику третьей функции принадлежности и вводим наименование терма, например, *высокий* в поле **Name** и нажмем **<Enter>**. В результате получим графическое окно, изображенное на рис. 4.1.3.

Шаг 11. Зададим функции принадлежности термов для переменной **x2**. Для лингвистической оценки этой переменной будем использовать 5 термов с гауссовыми функциями принадлежности. Для этого активизируем переменную **x2** с помощью щелчка левой кнопки мыши на блоке **x2.** Зададим диапазон изменения переменной **x2**. Для этого введем **-4.4 1.7** в поле **Range** (рис. 4.1.4) и нажмем **<Enter>**. Затем в меню **Edit** выберем команду **Add MFs...**. В появившимся диалоговом окне выбираем тип функции принадлежности **gaussmf** в поле **MF type** и **5** термов в поле **Number of MFs**. После этого нажимаем **<Enter>.**

Шаг 12. По аналогии с *шагом 10* зададим следующие наименования термов переменной $\mathbf{x_2}$: *низкий, ниже среднего, средний, выше среднего, высокий.* В результате получим графическое окно, изображенное на рис. 4.1.4.

Шаг 13. Зададим функции принадлежности переменной **y**. Для лингвистической оценки этой переменной будем использовать 5 термов с треугольными функциями принадлежности. Для этого активизируем переменную **y** с помощью щелчка левой кнопки мыши на блоке **y.** Зададим диапазон изменения переменной **y**. Для этого введем **-50 50** в поле **Range** (рис. 4.1.5) и нажмем **<Enter>**. Затем в меню **Edit** выберем команду **Add MFs...**. В появившимся диалоговом окне выбираем **5** термов в поле **Number of MFs**. После этого нажимаем **<Enter>.**

Шаг 14. По аналогии с *шагом 10* зададим следующие наименования термов переменной **y**: *низкий, ниже среднего, средний, выше среднего, высокий.* В результате получим графическое окно, изображенное на рис. 4.1.5.

Шаг 15. Перейдем в редактор базы знаний **RuleEditor**. Для этого в меню **Edit** выберем команду **Edit rules...**.

Шаг 16. На основе визуального наблюдения за графиком, изображенным на рис. 4.1.1, сформулируем следующие девять правил.

1. Если x1=*средний*, то y=*средний*.
2. Если x1=*низкий* и x2=*низкий*, то y=*высокий*.
3. Если x1=*низкий* и x2=*высокий*, то y=*высокий*.
4. Если x1=*высокий* и x2=*высокий*, то y=*выше среднего*.

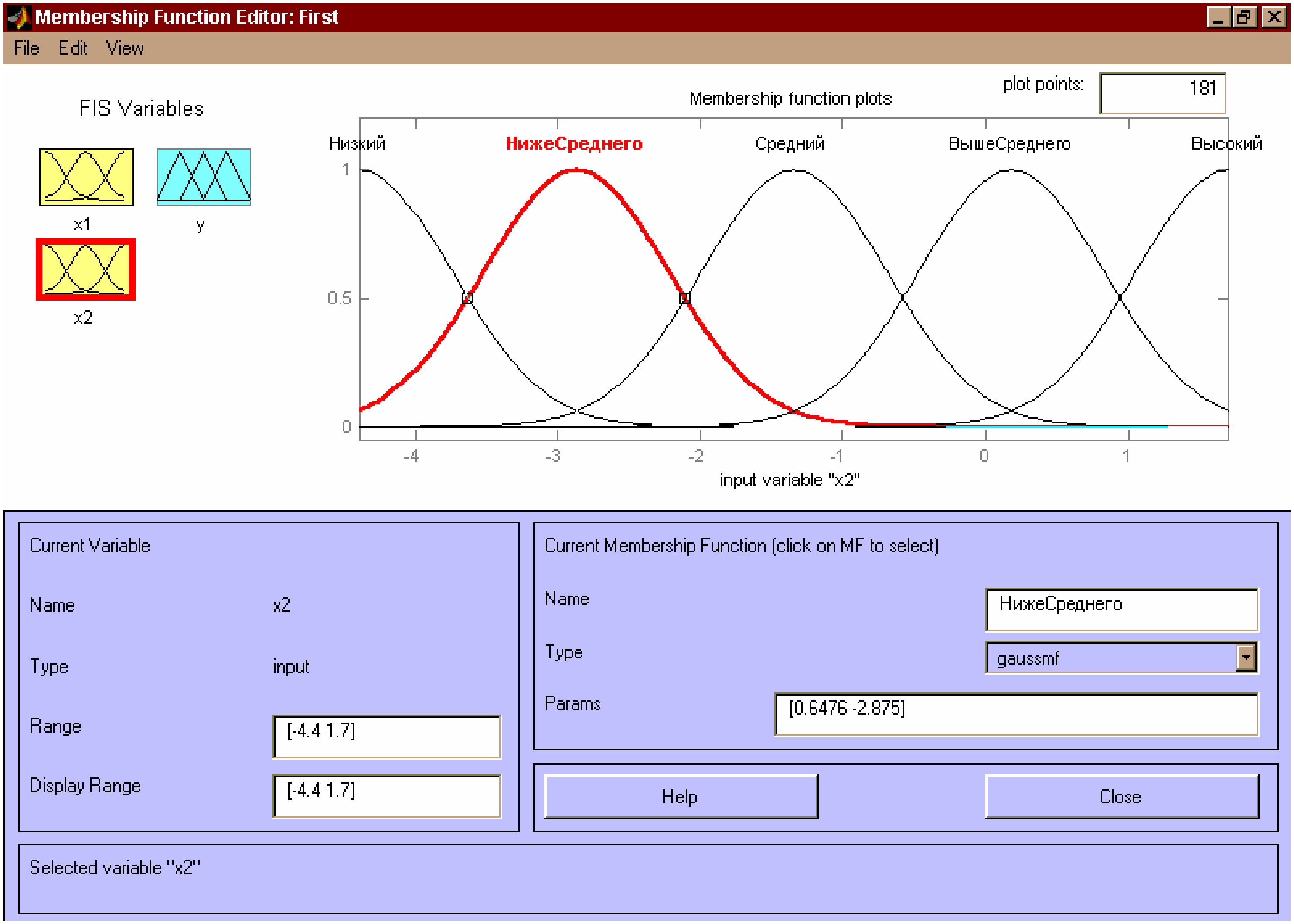

Рис. 4.1.4. Функции принадлежности переменной x2

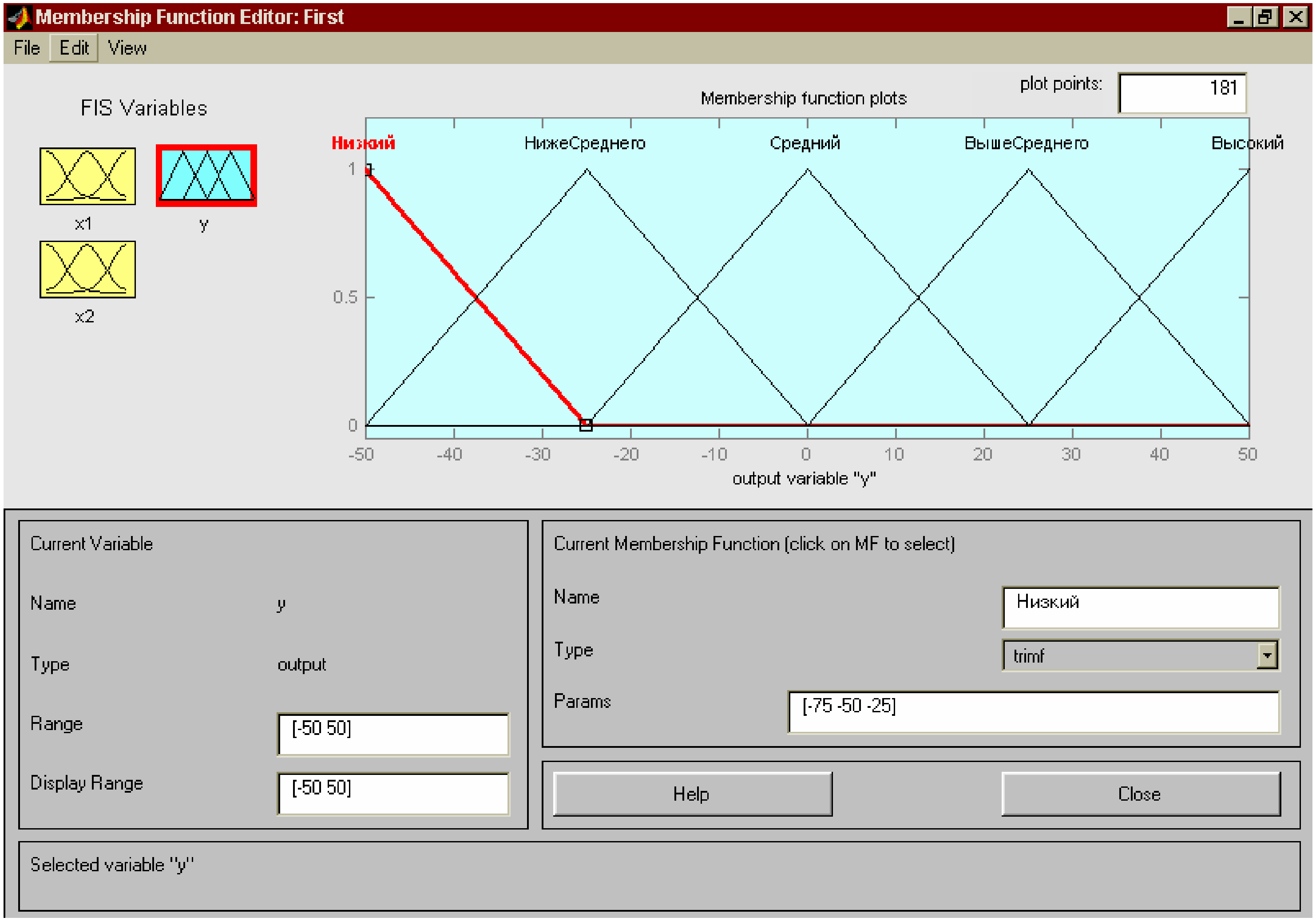

Рис. 4.1.5. Функции принадлежности переменной y

5. Если x1=*высокий* и x2=*низкий*, то y=*выше среднего*.
6. Если x1=*высокий* и x2=*средний*, то y=*средний*.
7. Если x1=*низкий* и x2=*средний*, то y=*низкий*.
8. Если x1=*высокий* и x2=*выше среднего*, то y=*средний*.
9. Если x1=*высокий* и x2=*ниже среднего*, то y=*средний*.

Для ввода правила необходимо выбрать в меню соответствующую комбинацию термов и нажать кнопку **Add rule**. На рис. 4.1.6 изображено окно редактора базы знаний после ввода всех девяти правил. Число, приведенное в скобках в конце каждого правила, представляет собой весовой коэффициент соответствующего правила.

Шаг 17. Сохраним созданную систему. Для этого в меню **File** выбираем в подменю **Export** команду **To disk**.

На рис. 4.1.7 приведено окно визуализации нечеткого логического вывода. Это окно активизируется командой **View rules...** меню **View.** В поле **Input** указываются значения входных переменных, для которых выполняется логический вывод. На рис. 4.1.8 приведена поверхность “входы-выход”, соответствующая синтезированной нечеткой системе. Для вывода этого окна необходимо использовать команду **View surface...** меню **View.** Сравнивая поверхности на рис. 4.1.1 и на рис. 4.1.8, можно сделать вывод, что нечеткие правила достаточно хорошо описывают сложную нелинейную зависимость.

4.2. Лабораторная работа №2

Проектирование нечетких систем Суджено

Рассмотрим основные этапы проектирования систем типа Суджено на примере создания системы нечеткого логического вывода, моделирующей зависимость

$$y = x_1^2 \cdot \sin(x_2 - 1),\ x_1 \in [-7,3],\ x_2 \in [-4.4,1.7],$$

график которой представлен на рис. 4.1.1. Моделирование этой зависимости будем осуществлять с помощью следующей базы правил.

1. Если x1=*средний*, то y=0.
2. Если x1=*высокий* и x2=*высокий*, то y=2x1+2x2+1.
3. Если x1=*высокий* и x2=*низкий*, то y=4x1-x2.
4. Если x1=*низкий* и x2=*средний*, то y=8x1+2x2+8.
5. Если x1=*низкий* и x2=*низкий*, то y=50.
6. Если x1=*низкий* и x2=*высокий*, то y=50.

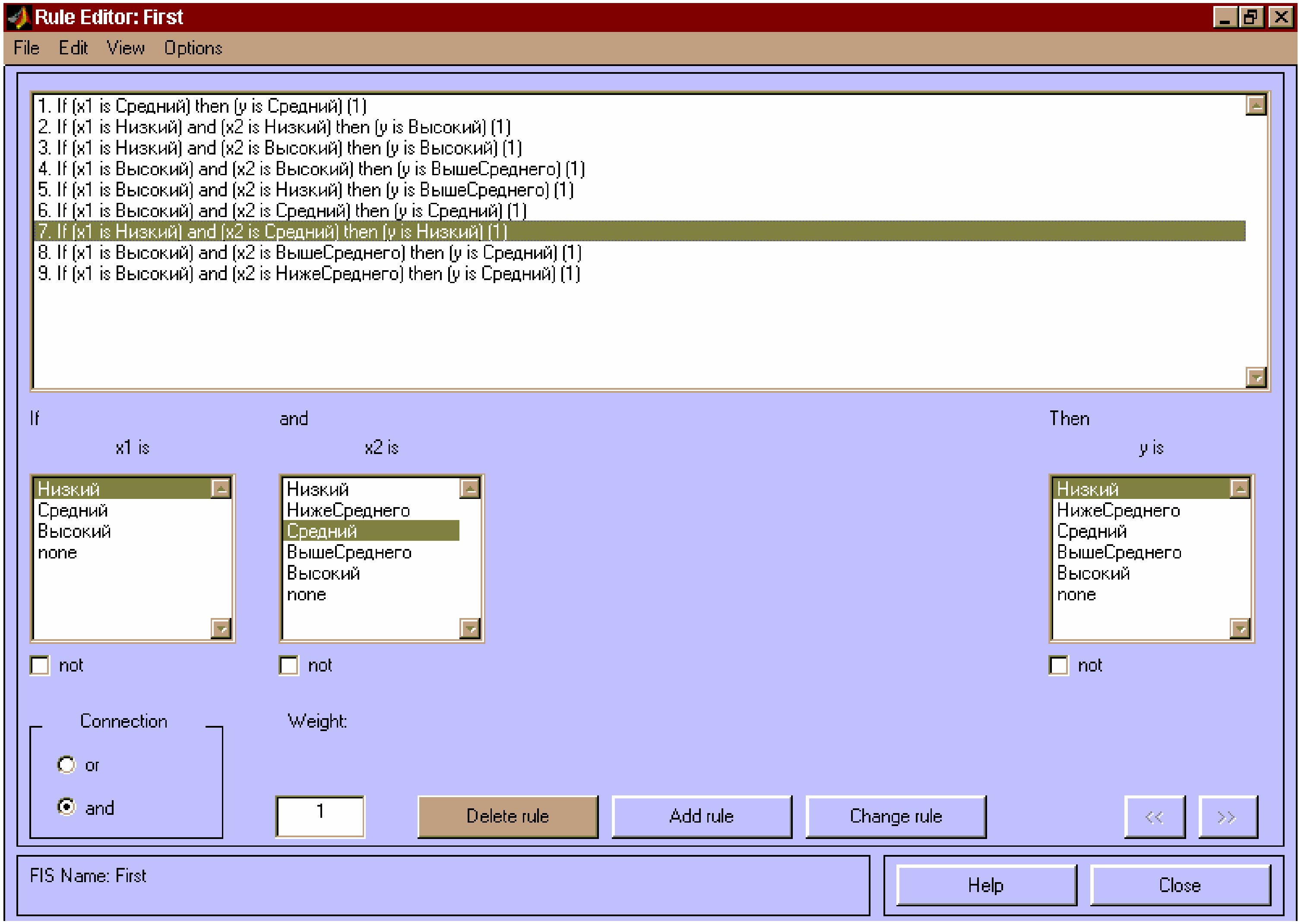

Рис. 4.1.6. База знаний в RuleEditor

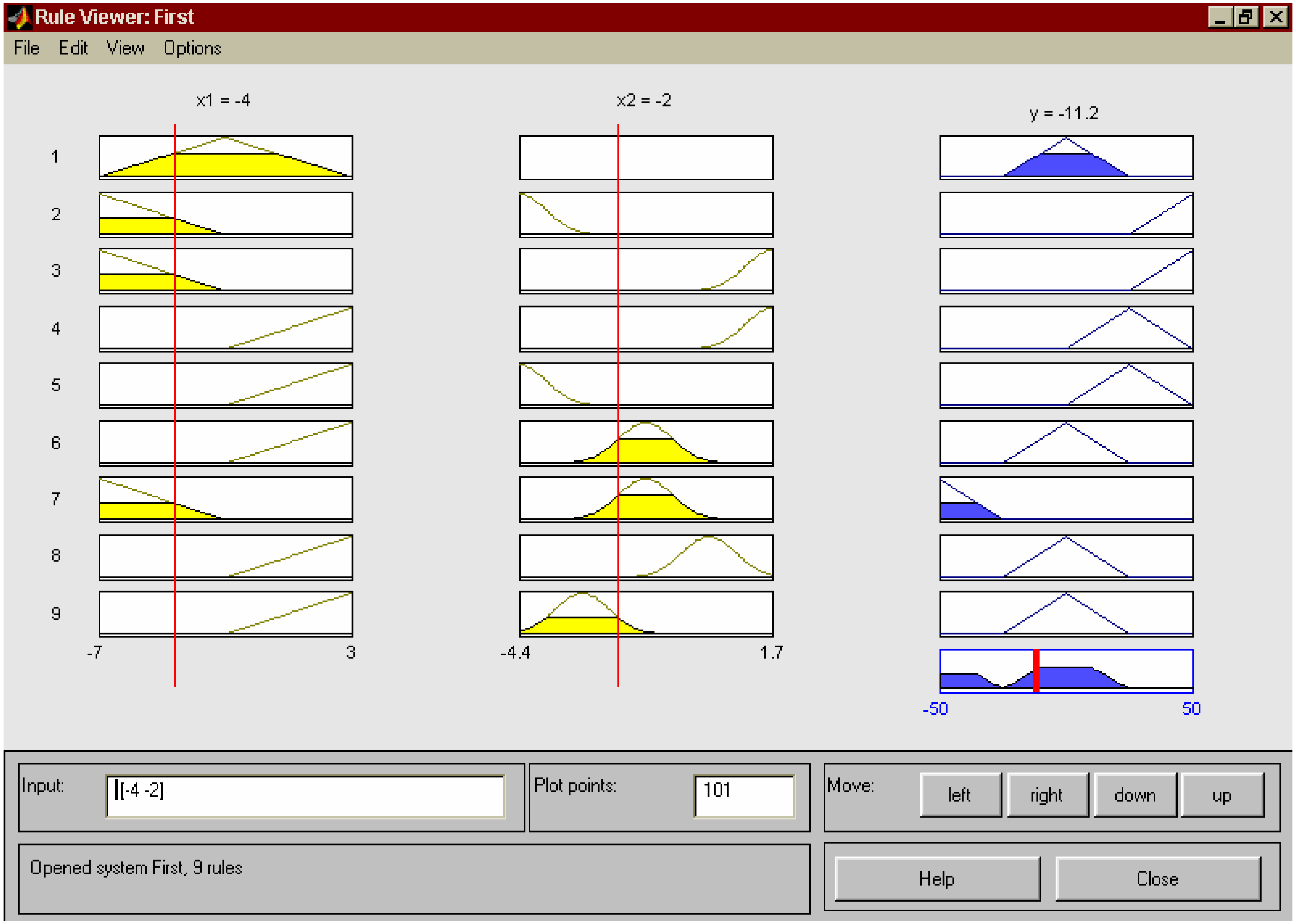

Рис. 4.1.7. Визуализация нечеткого логического вывода в RuleViewer

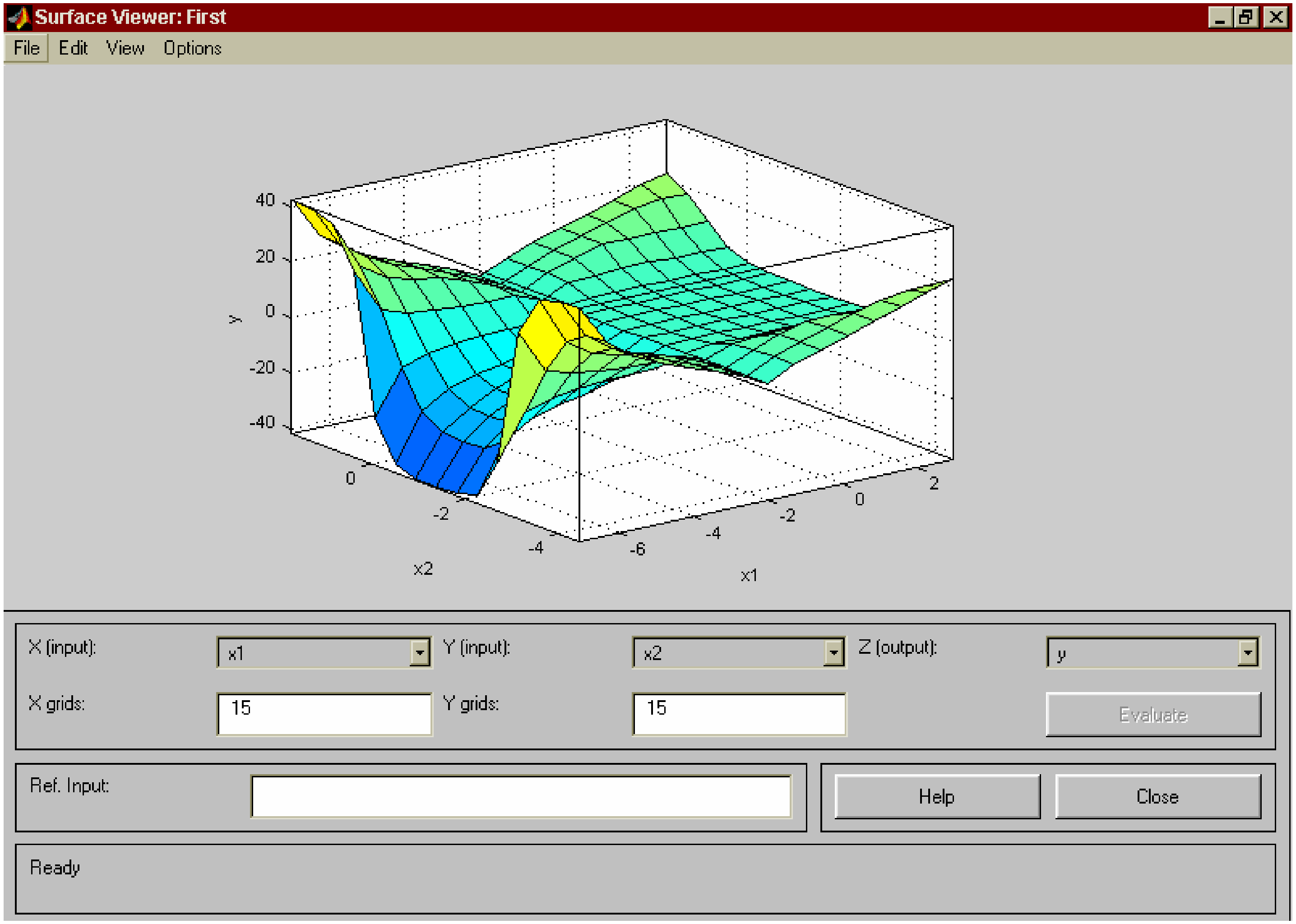

Рис. 4.1.8. Поверхность “входы-выход” в окне SurfaceViwer

Проектирование нечеткой системы осуществляется в следующей последовательности.

Шаг 1. Для загрузки основного **fis**-редактора введем слова **fuzzy** в командной строке. После этого откроется новое графическое окно, показанное на рис. 4.1.2.

Шаг 2. Выберем тип системы. Для этого в меню **File** выбираем в подменю **New fis…** команду **Sugeno**.

Шаг 3. Добавим вторую входную переменную. Для этого в меню **Edit** выбираем команду **Add input**.

Шаг 4. Переименуем первую входную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке **input1**, введем новое обозначение **x1** в поле редактирования имени текущей переменной и нажмем **<Enter>**.

Шаг 5. Переименуем вторую входную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке **input2**, введем новое обозначение **x2** в поле редактирования имени текущей переменной и нажмем **<Enter>**.

Шаг 6. Переименуем выходную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке **output1**, введем новое обозначение **y** в поле редактирования имени текущей переменной и нажмем **<Enter>**.

Шаг 7. Зададим имя системы. Для этого в меню **File** выбираем в подменю **Export** команду **To disk** и введем имя файла, например, **FirstSugeno**.

Шаг 8. Перейдем в редактор функций принадлежности. Для этого сделаем двойной щелчок левой кнопкой мыши на блоке **x1**.

Шаг 9. Зададим диапазон изменения переменной **x1**. Для этого введем **-7 3** в поле **Range** (рис. 4.2.1) и нажмем **<Enter>**.

Шаг 10. Зададим функции принадлежности переменной **x1**. Для лингвистической оценки этой переменной будем использовать 3 терма с треугольными функциями принадлежности, которые установлены по умолчанию. Зададим наименования термов переменной **x1**. Для этого делаем один щелчок левой кнопкой мыши по графику первой функции принадлежности (рис. 4.2.1). Затем введем наименование терма **Низкий** в поле **Name**. Затем делаем один щелчок левой кнопкой мыши по графику второй функции принадлежности и вводим наименование терма **Средний** в поле **Name**. Еще раз делаем один щелчок левой кнопкой мыши по графику третьей функции принадлежности и вводим наименование терма **Высокий** в поле **Name,** нажимаем **<Enter>**. В результате получим графическое окно, изображенное на рис. 4.2.1.

Шаг 11. Зададим функции принадлежности переменной **x2**. Для лингвистической оценки этой переменной будем использовать 3 терма с треугольными функциями принадлежности, которые установлены по умолчанию. Для этого активизируем переменную **x2** с помощью щелчка левой кнопки мыши на блоке **x2.** Зададим диапазон изменения переменной **x2**. Для этого введем **-4.4 1.7** в поле **Range** (рис. 4.2.2) и нажмем **<Enter>**. По аналогии с предыдущим шагом зададим следующие наименования термов переменной **x2**: **Низкий**, **Средний**, **Высокий**. В результате получим графическое окно, изображенное на рис. 4.2.2.

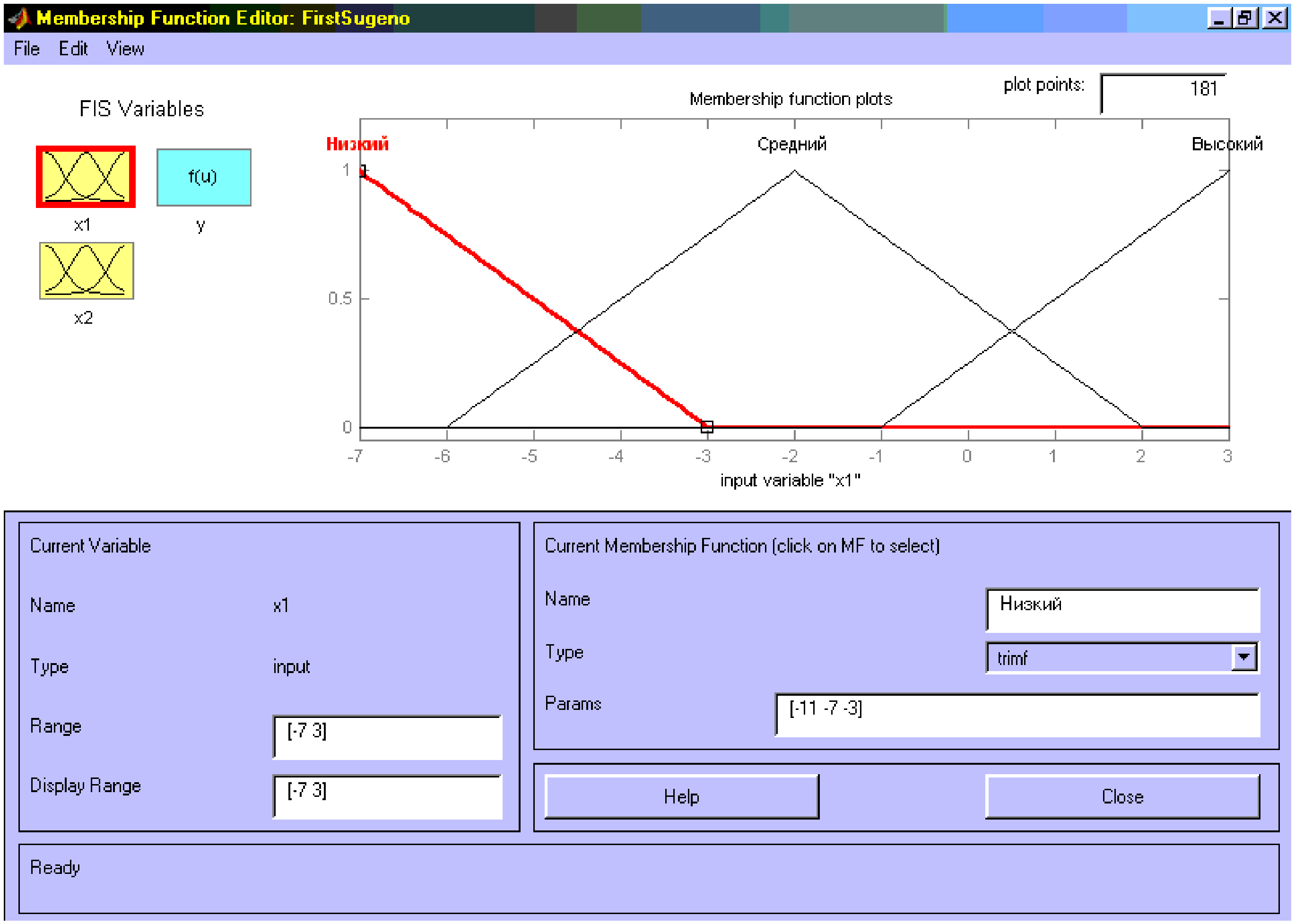

Рис. 4.2.1. Функции принадлежности переменной x1

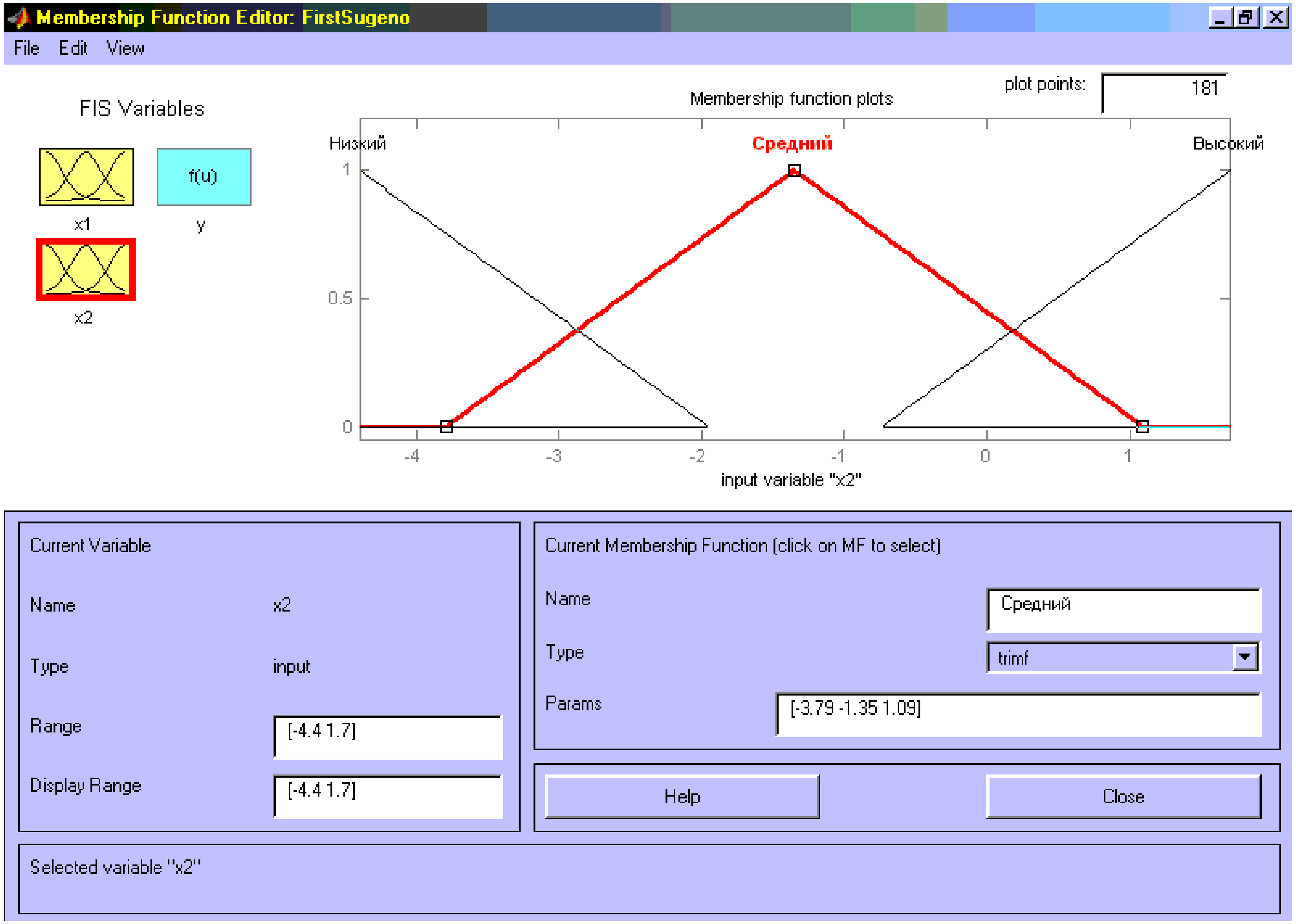

Рис. 4.2.2. Функции принадлежности переменной x2

Шаг 12. Зададим линейные зависимости между входами и выходом, приведенные в базе знаний. Для этого активизируем переменную **y** с помощью щелчка левой кнопки мыши на блоке **y.** В правом верхнем углу появилось обозначение трех функций принадлежности, каждая из которых соответствует одной линейной зависимости между входами и выходам. В базе знаний приведены 5 различных зависимостей: y=50; y=4x1-x2; y=2x1+2x2+1; y=8x1+2x2+8; y=0. Поэтому добавим еще две зависимости путем выбора команды **Add Mfs…** меню **Edit.** В появившимся диалоговом окне в поле **Number of MFs** выбираем 2 и нажимаем кнопку **<Enter>**.

Шаг 13. Зададим наименования и параметры линейных зависимостей. Для этого делаем один щелчок левой кнопкой мыши по наименованию первой зависимости **mf1**. Затем вводим наименование зависимости, например, **50** в поле **Name** и устанавливаем тип зависимости – константа – путем выбора опции **Constant** в меню **Type**. После этого вводим значение параметра **50** в поле **Params**. Аналогично для второй зависимости **mf2** введем наименование зависимости, например, **8+8x1+2x2**. Затем укажем линейный тип зависимости путем выбора опции **Linear** в меню **Type** и введем параметры зависимости **8 2 8** в поле **Params**. Для линейной зависимости порядок параметров следующий: первый параметр – коэффициент при первой переменной, второй – при второй и т.д., и последний параметр – свободный член зависимости. Аналогично для третьей зависимости **mf3** введем наименование зависимости, например, **1+2x1+2x2**, укажем линейный тип зависимости и введем параметры зависимости **2 2 1**. Для четвертой зависимости **mf4** введем наименование зависимости, например, **4x1-x2**, укажем линейный тип зависимости и введем параметры зависимости **4 -1 0**. Для пятой зависимости **mf5** введем наименование зависимости, например, **0**, укажем тип зависимости – константа и введем параметр зависимости **0**. В результате получим графическое окно, изображенное на рис. 4.2.3.

Шаг 14. Перейдем в редактор базы знаний **RuleEditor**. Для этого выберем в меню **Edit** команду **Edit rules...**. и введем правила базы знаний. Для ввода правила необходимо выбрать соответствующую комбинацию термов и зависимостей и нажать кнопку **Add rule**. На рис. 4.2.4 изображено окно редактора базы знаний после ввода всех шести правил. На рис. 4.2.5 приведено окно визуализации нечеткого логического вывода. Это окно активизируется командой **View rules...** меню **View.** В поле **Input** указываются значения входных переменных, для которых выполняется логический вывод. Из этого рисунка видно, что значение выходной переменной рассчитывается как среднее взвешенное значение результатов вывода по каждому правилу. На рис. 4.2.6 приведена поверхность “входы-выход”, соответствующая синтезированной нечеткой системе. Для вывода этого окна необходимо использовать команду **View surface...** меню **View.**

Сравнивая поверхности на рис. 4.1.1, рис. 4.1.8 и на рис. 4.2.6 можно сделать вывод, что нечеткие правила достаточно хорошо описывают сложную нелинейную зависимость. При этом модель типа Суджено более точная.

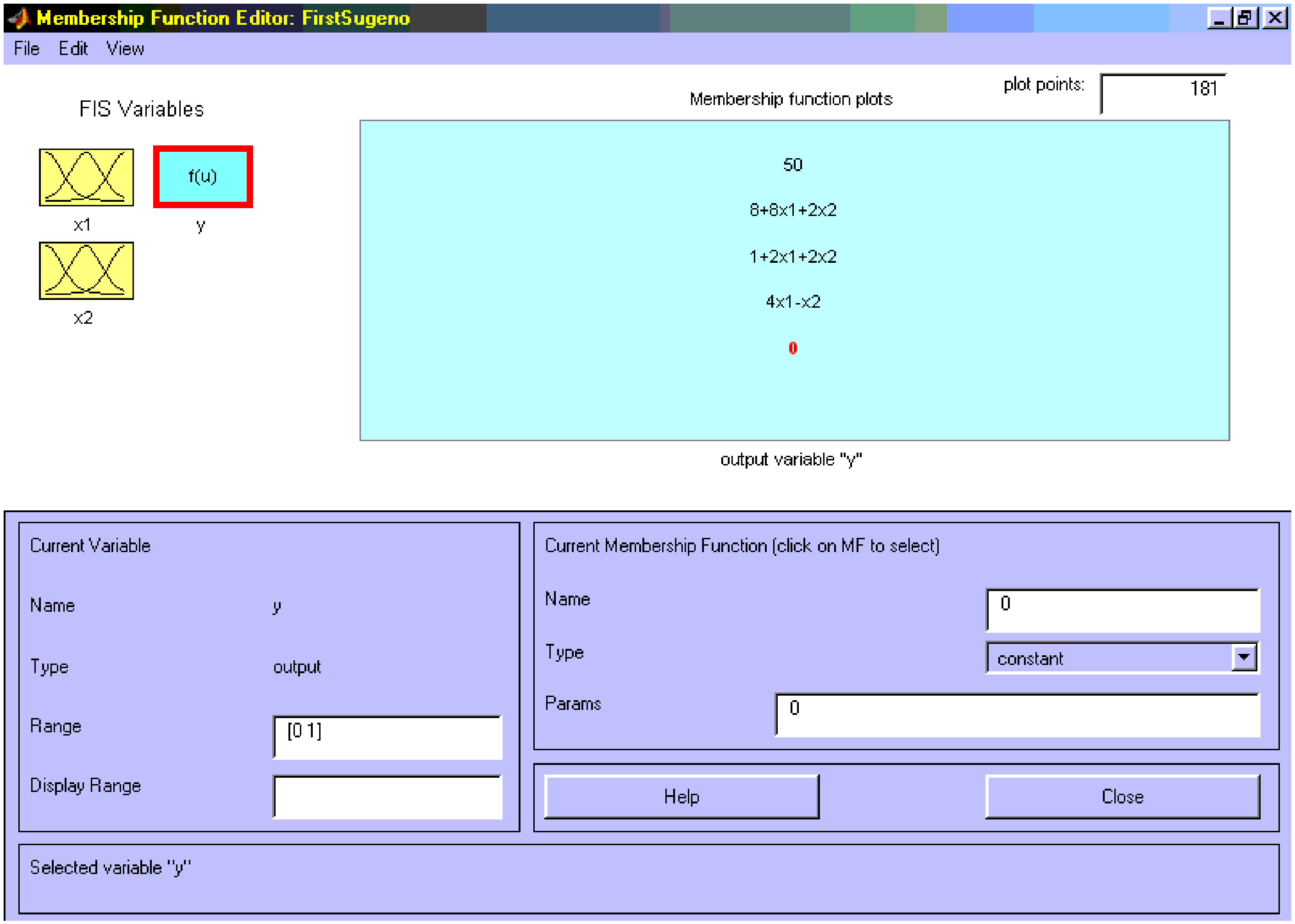

Рис. 4.2.3. Окно линейных зависимостей “входы-выход”

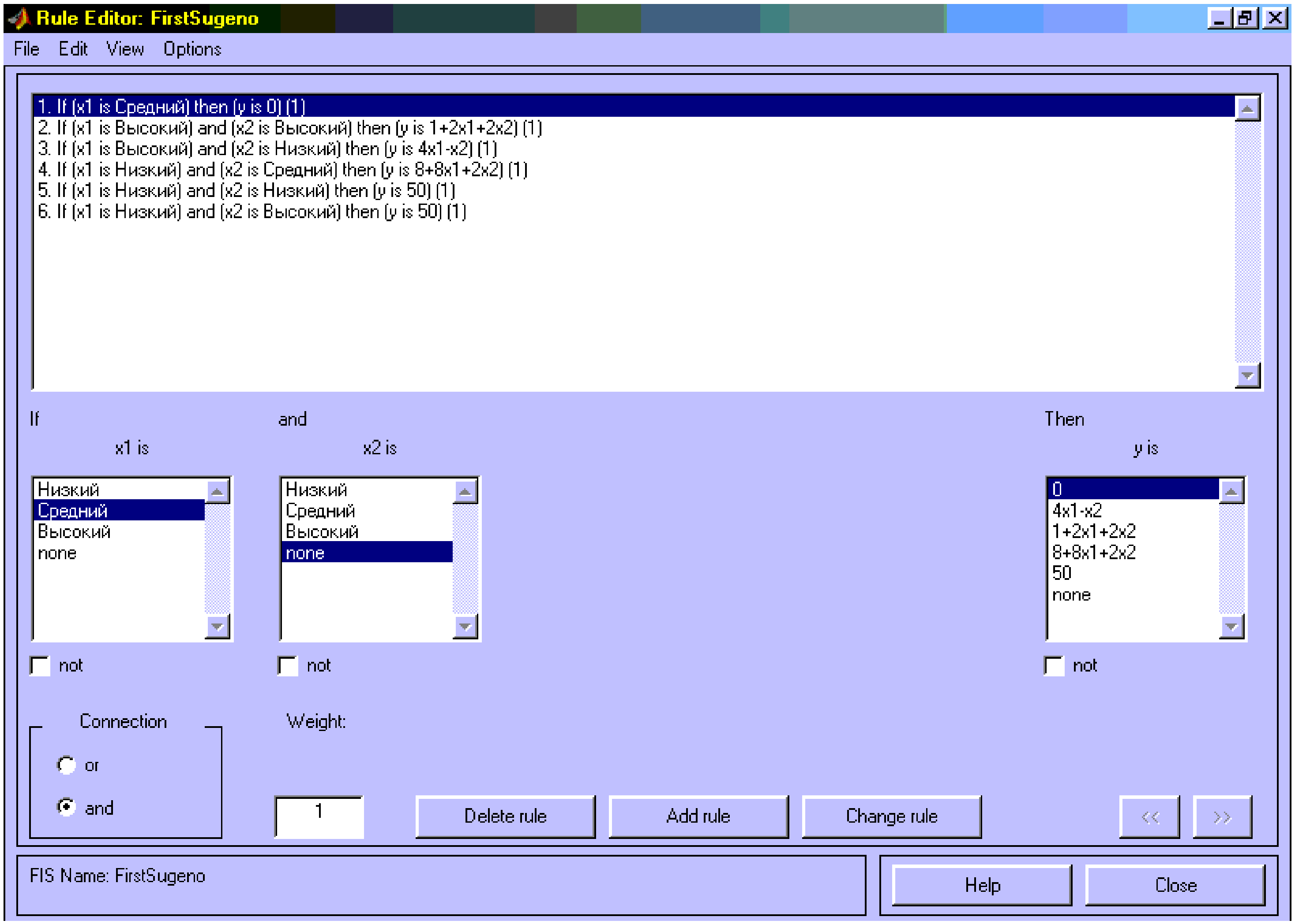

Рис 4.2.4. Нечеткая база знаний для системы Суджено

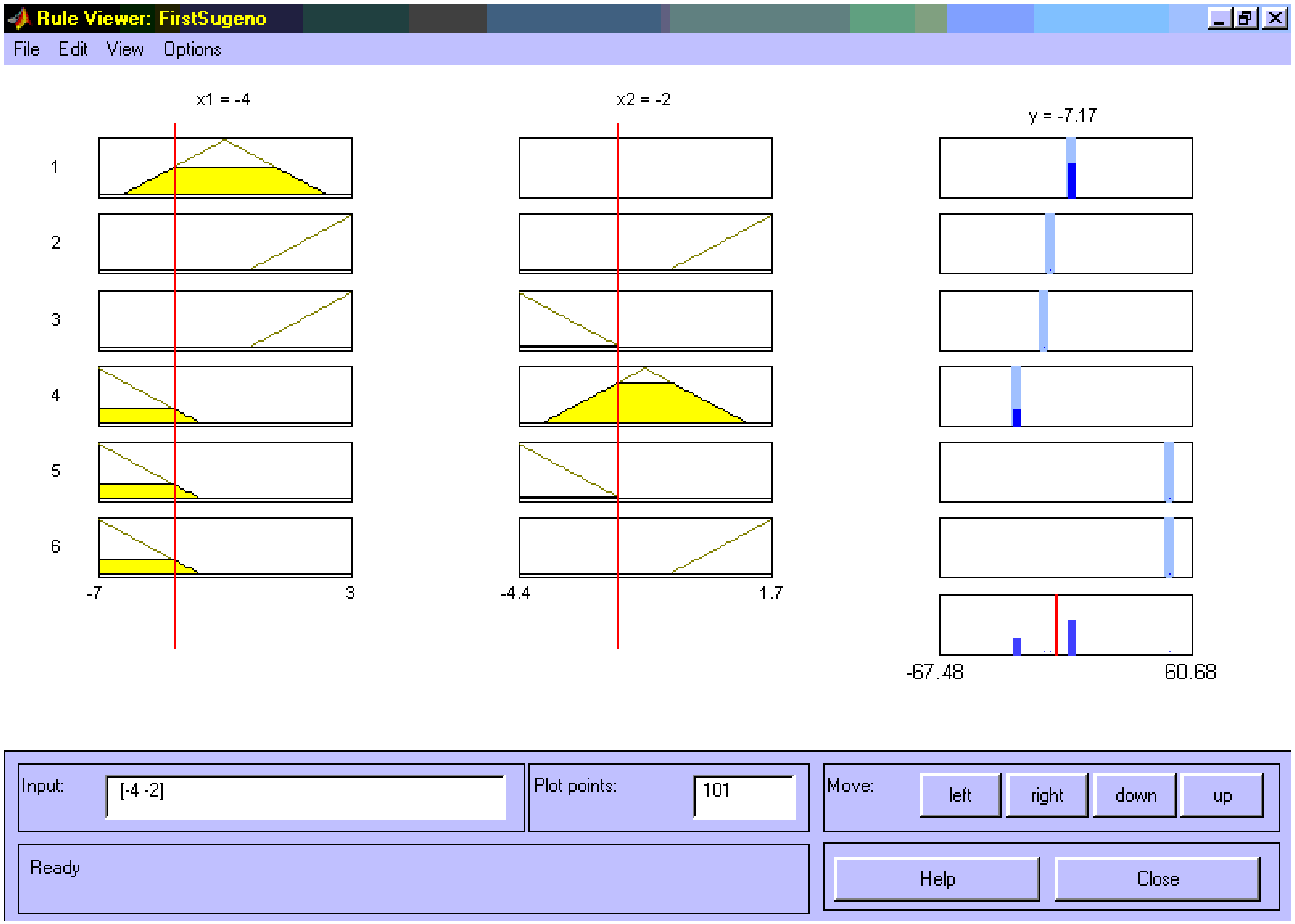

Рис. 4.2.5. Визуализация нечеткого логического вывода для системы типа Суджено

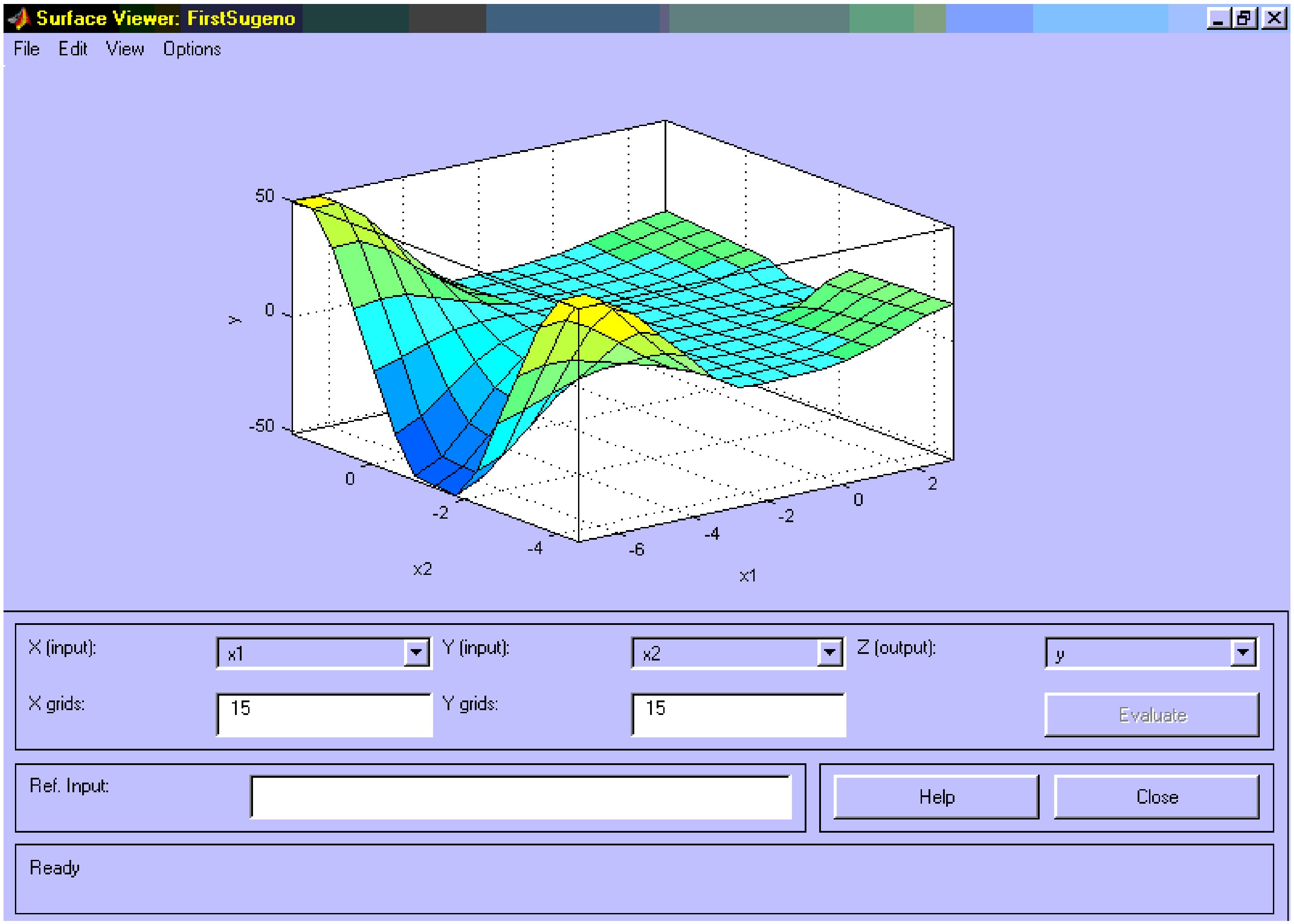

Рис. 4.2.6. Поверхность “входы-выход” для системы типа Суджено

4.3. Лабораторная работа №3
<u>Метод матричного шаблона</u>
(для решения нечетких реляционных уравнений)

Для нахождения максимального и минимальных решений нечетких реляционных уравнений в среде *MatLab* разработана программа[7] **MMSH.m**. Файл **MMSH.m** является внешней, или m-функцией. Выполнив команду help MMSH, можно ознакомиться с описанием и структурой функции. Эта функция позволяет решить нечеткое реляционное уравнение $A \circ r = b$ с максминной или максмультипликативной композициями. Чтобы воспользоваться ею, необходимо выполнить следующие действия.

1. Открыть рабочую среду *MatLab* и скопировать файл **MMSH.m** в рабочую директорию системы *MatLab* (обычно **C:\MATLAB\work**). Если это невозможно (например, доступ к этой папке заблокирован), необходимо скопировать файл в любую доступную директорию, после чего, воспользовавшись одной из кнопок смены директории, указать путь к ней. Если все выполнено правильно, в окне рабочей директории будет отображен список файлов, включая **MMSH.m**.

<u>Примечание.</u> В случае недоступности рабочей директории можно также воспользоваться командой **File → Set Path**, чтобы добавить свою директорию в список используемых. Теперь функция готова к использованию.

2. Работа с системой *MatLab* в режиме прямых вычислений носит диалоговый характер. Пользователь набирает на клавиатуре вычисляемое выражение, редактирует его (если нужно) в командной строке и завершает ввод нажатием клавиши **<Enter>**, причем:

- если не указана переменная для значения результата вычислений, то ***MatLab*** назначает такую переменную с именем **ans**;
- знаком присваивания является знак равенства =;
- результат вычислений выводится в строках вывода (без знака »).

<u>Подготовка данных.</u>

Переменные и присваивание им значений. В системе *MatLab* можно задавать переменным определенные значения по правилу

Имя_переменной = Выражение **.**

Функции — это имеющие уникальные имена объекты, выполняющие определенные преобразования своих аргументов и при этом возвращающие результаты этих преобразований. Если функция возвращает несколько значений, то она записывается в виде

[Yl, Y2, …] = func(Xl, X2, …),

где Yl, Y2,... — список *выходных* параметров и XI, X2, ... — список *входных* аргументов (параметров).

Формирование векторов и матриц. Если надо задать вектор из нескольких элементов, то их значения следует перечислить в квадратных скобках, разделяя пробелами. Так, например, присваивание `V=[l 2 3]` (или

[7] Авторы программы: Желтова Е., Мануковская Г.

`V=[l,2,3]`) задает вектор *V*, имеющий три элемента со значениями *1*, *2* и *3*. После ввода вектора система выводит его на экран дисплея. Таким образом, векторы задаются списком своих элементов, разделяемых пробелами или запятыми. Список заключается в квадратные скобки.

```
>> V=[1 2 3]
V =
1     2     3
```

Задание матрицы требует указания нескольких строк. Для разграничения строк используется знак «*;*». Так, ввод `M=[1 2 3; 4 5 6; 7 8 9]` задает квадратную матрицу

$$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}.$$

Чтобы впоследствии вывести на экран значение переменной, необходимо написать ее имя и нажать **<Enter>**.

3. Рассмотрим, как воспользоваться функцией **MMSH.m** на примере. Решим нечеткое реляционное уравнение

$$\begin{pmatrix} 0{,}9 & 0{,}5 & 0{,}4 & 0{,}4 & 0{,}9 \\ 0{,}2 & 0{,}3 & 0 & 0{,}6 & 0{,}2 \\ 0{,}6 & 0{,}7 & 1 & 0{,}7 & 0{,}1 \end{pmatrix} \circ r = \begin{pmatrix} 0{,}72 \\ 0{,}3 \\ 0{,}9 \end{pmatrix},$$

где $\circ$ – максминная композиция. Ниже представлен ввод данных.

```
>> A=[0.9 0.5 0.4 0.4 0.9; 0.2 0.3 0 0.6 0.2; 0.6 0.7 1 0.7
0.1]

A =

    0.9000    0.5000    0.4000    0.4000    0.9000
    0.2000    0.3000         0    0.6000    0.2000
    0.6000    0.7000    1.0000    0.7000    0.1000

>> B=[0.72; 0.3; 0.9]

B =

    0.7200
    0.3000
    0.9000
```

Пусть результаты вычислений записываются в переменные *`Rmax`*, *`Rmin`*. Если используется максминная композиция, то третий параметр полагается равным 0; если максмультипликативная – 1. Ниже представлены выходные переменные `Rmax, Rmin`, которые содержат максимальное и минимальные решения нечеткого реляционного уравнения.

```
>> [Rmax, Rmin] = MMSH(A,B,0)

Rmax =

    0.7200
    1.0000
    0.9000
    0.3000
    0.7200


Rmin =

    0.7200    0.7200         0         0
    0.3000         0    0.3000         0
    0.9000    0.9000    0.9000    0.9000
         0    0.3000         0    0.3000
         0         0    0.7200    0.7200
```

Ответ:

$$R_{max} = \begin{pmatrix} 0{,}72 \\ 1 \\ 0{,}9 \\ 0{,}3 \\ 0{,}72 \end{pmatrix}, \quad R_{min} = \begin{pmatrix} 0{,}72 & 0{,}72 & 0 & 0 \\ 0{,}3 & 0 & 0{,}3 & 0 \\ 0{,}9 & 0{,}9 & 0{,}9 & 0{,}9 \\ 0 & 0{,}3 & 0 & 0{,}3 \\ 0 & 0 & 0{,}72 & 0{,}72 \end{pmatrix}.$$

4.4. Лабораторная работа №4
Нечеткая кластеризация

Нечеткая кластеризация данных в среде *MatLab* осуществляется с помощью расширения *Fuzzy Logic Toolbox*. В этот пакет входит программа **Clustering**, которая является некоторой визуализированной средой для осуществления классификации данных с помощью алгоритмов *Fuzzy C-means* and *Subtractive clustering*.

1. Для вызова программы **Clustering** наберите команду **findcluster**, в результате чего появится новое окно.

2. Перейдите в новое окно и нажмите кнопку загрузки данных **Load Data** (данные загружаются в формате ***.dat***; формат данных: матрица объект-признак, столбцы соответствуют признакам, строки – объектам, числа разделяются пробелом, дробная часть отделяется точкой).

Замечание. Файл *.dat можно подготовить средствами *MatLab*. Как один из способов можно предложить следующий:

– В командной строке набрать >> data = []; (в рабочем пространстве будет создана переменная data типа массив, с пустым содержимым).
– Открыть редактор массива (для этого переключится в панель Workspace и кликнуть дважды по переменной data).
– Указать размерность массива в полях Size и By с учетом пункта 2 задания – строка объекты, столбцы – признаки.
– После ввода всех данных закрыть редактор и в командной строке набрать

>> save <путь\имя файла.dat> data – ASCII

3. В окне программы появится двумерная визуализация данных, которая возможна только в случае выбора двух признаков, поэтому целесообразно комбинировать различные пары признаков, чтобы более полно проанализировать структуру данных.

4. Выбор алгоритма кластеризации осуществляется с помощью кнопки **Method**. Выберите метод **fcm** (алгоритм *Fuzzy C-means*).

5. Задайте параметры классификации для метода **fcm**: **Cluster num** – количество искомых нечетких кластеров; **Max iteration#** – максимальное количество итераций (по умолчанию – 100); **Min Improvement** – параметр сходимости алгоритма (по умолчанию – 0.00001); **Exponent** – экспоненциальный вес для расчета матрицы нечеткого разбиения (по умолчанию – 2). Если параметры специально не определять, то их значения будут выставлены по умолчанию.

6. Нажмите кнопку **Start**. На рис. 4.4.1 представлена классификация данных методом *Fuzzy C-means*. Черными точками изображены центры кластеров (их 5). Изменяя параметры **Cluster Num** и **Exponent**, можно увидеть, как изменилась классификация.

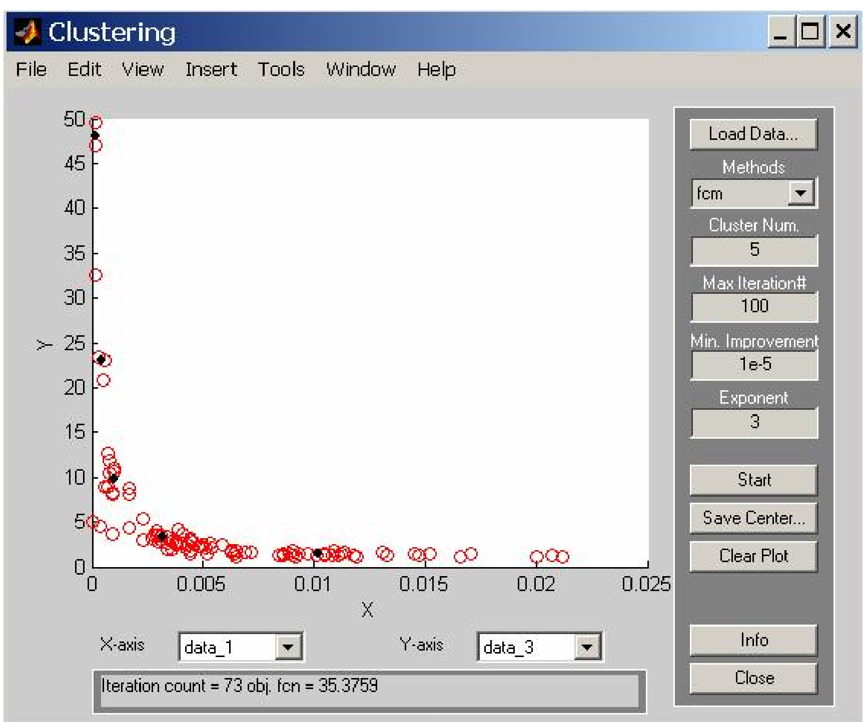

Рис. 4.4.1

7. Теперь с помощью кнопки **Method** выберите **Subtractive** (метод *Subtractive clustering*). Задайте следующие параметры для этого алгоритма: **Influence Range** – радиус (число из *[0,1]* (для нормализованных данных) или вектор, каждая компонента которого представляет собой радиус для соответствующей координаты данных; если задать одно число, то радиус для всех признаков будет считаться одинаковым); **Squash Factor** – *фактор подавления* – используется для расширения «сферы влияния» выделенного центра кластера для уменьшения потенциала точек, относящихся к данному кластеру (по умолчанию – 1.25); **Accept Ratio** – *коэффициент принятия* – устанавливает во сколько раз должен быть больше потенциал следующего кластера по отношению к первому, чтобы его можно было объявить центром нового кластера (по умолчанию – 0.5); **Reject Ratio** – *коэффициент отклонения* – устанавливает потенциал как часть от потенциала центра первого кластера: если потенциал точки ниже коэффициента отклонения, то элемент данных не будет объявляться центром класса (по умолчанию равен *0.15*).

8. Нажмите кнопку **Start.** На рис. 4.4.2 представлена классификация данных методом *Subtractive clustering.*

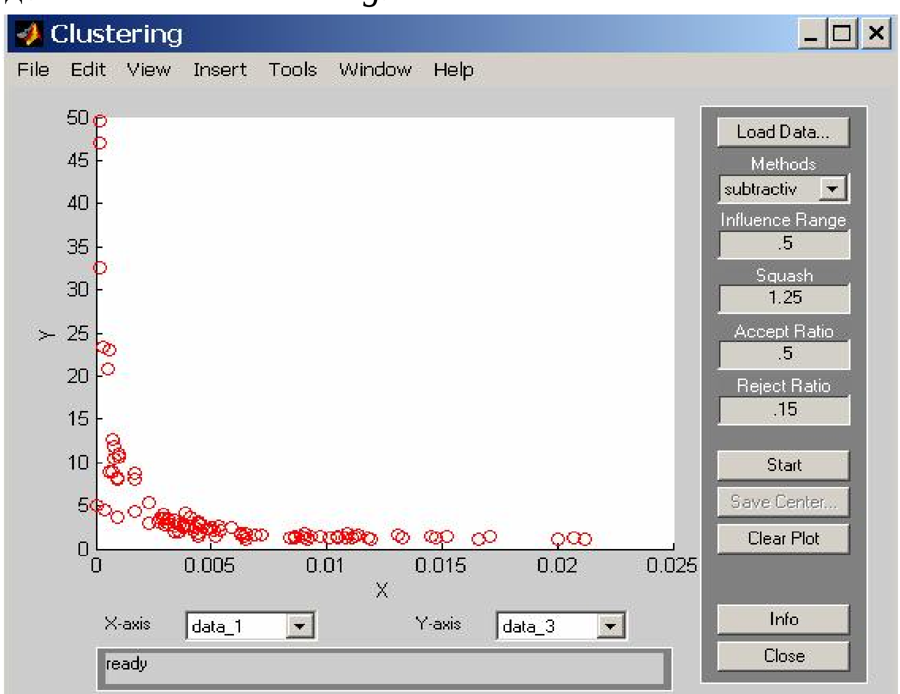

Рис. 4.4.2

Отметим некоторые <u>возможности, предоставляемые средой</u> **<u>Clustering.</u>**

1. Для более детального изучения структуры данных можно осуществить вращение данных в трехмерной системе координат с помощью функции **Tools/ Rotate 3D** и **Tools/ Move Camera** (рис. 4.4.3).

2. С помощью **Tools/Basic Fittings** можно выбрать подходящую зависимость для данных или для центров кластеров. Для этого в **Select data** выбрать метод подбора зависимости. В результате выводится вид зависимости (уравнение и график), значения коэффициентов, норма остатков, а также прогноз одного или нескольких значений. В этом режиме можно подобрать вид зависимости, которая наилучшим образом аппроксимирует данные (рис. 4.4.4 и 4.4.5).

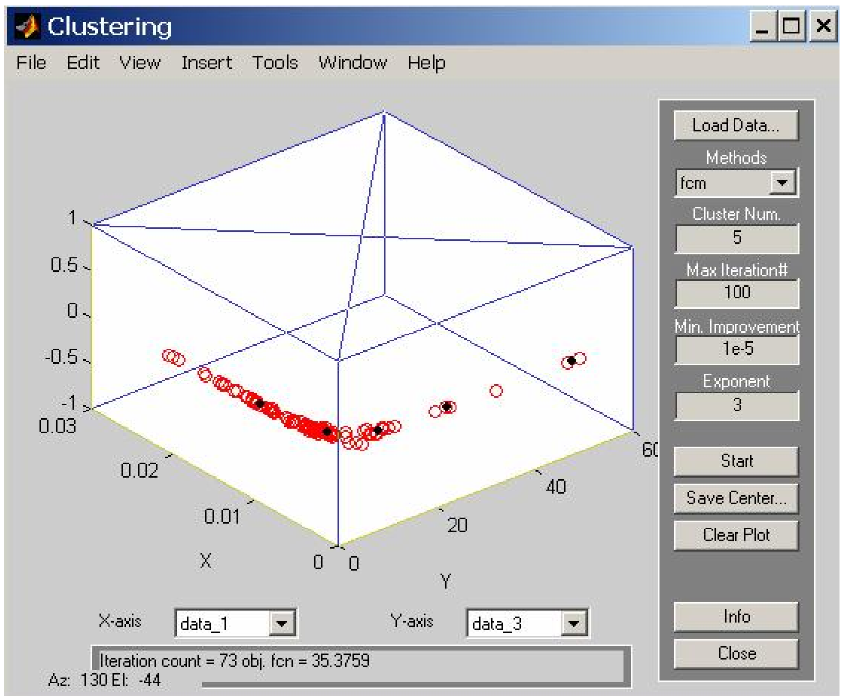

Рис. 4.4.3

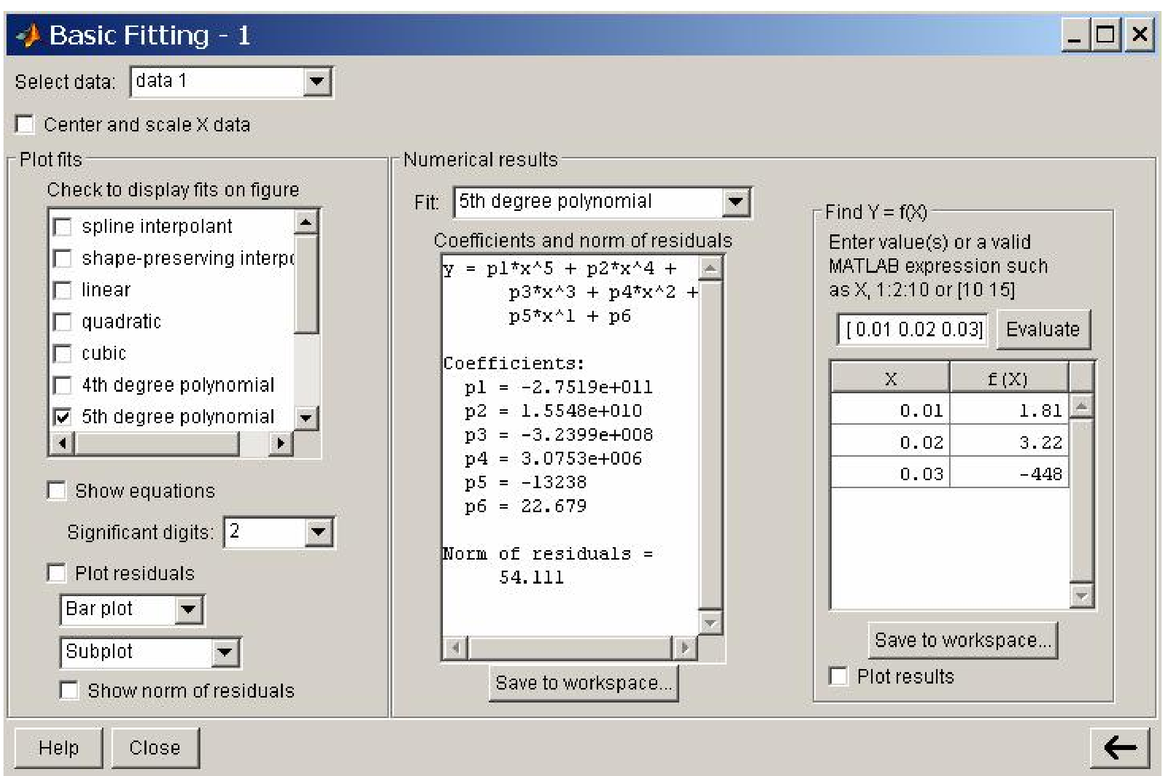

Рис. 4.4.4

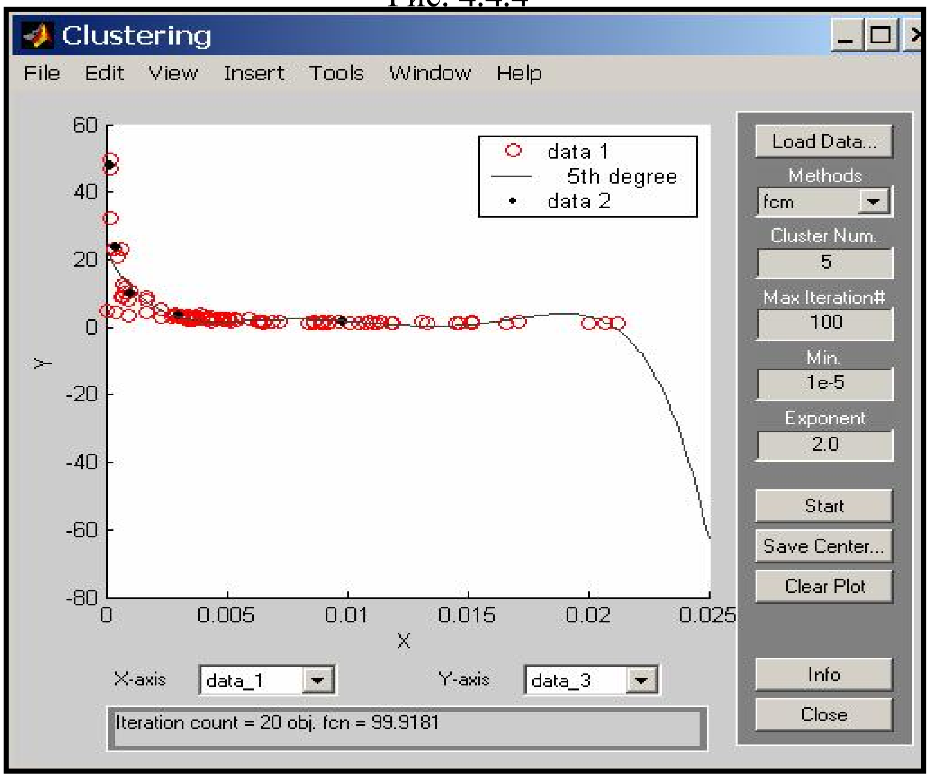

Рис. 4.4.5

3. Расчет основных статистик с помощью **Tools/Data Statistics**: минимальное и максимальное значения, среднее, медиана, среднеквадратическое [стандартное] отклонение, диапазон изменения (max-min) для каждой координаты. Значения статистик можно представить графически (рис. 4.4.6 и 4.4.7).
4. Редактирование изображений с помощью пунктов меню **Insert/ X,Y,Z Label**, **Title**, **Legend**, **Colorbar**, **Arrow**, **Text**, **Line**, **Axes**, **Light**.

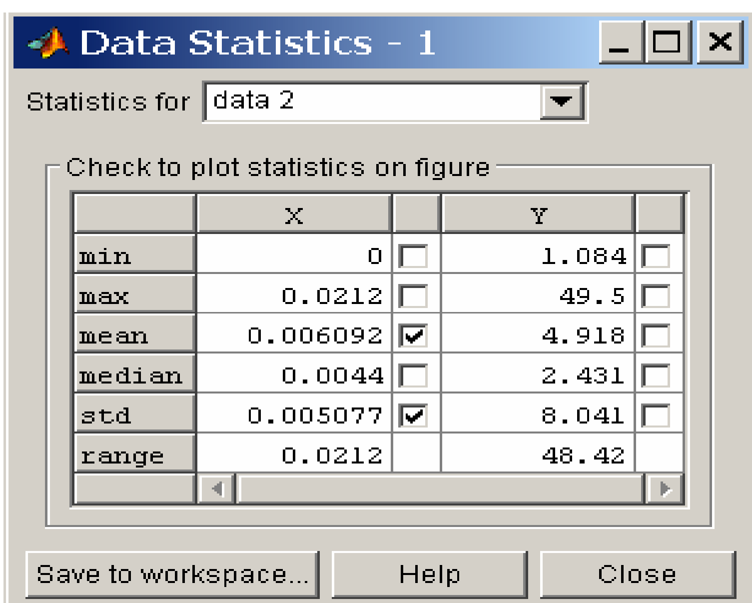

Рис. 4.4.6

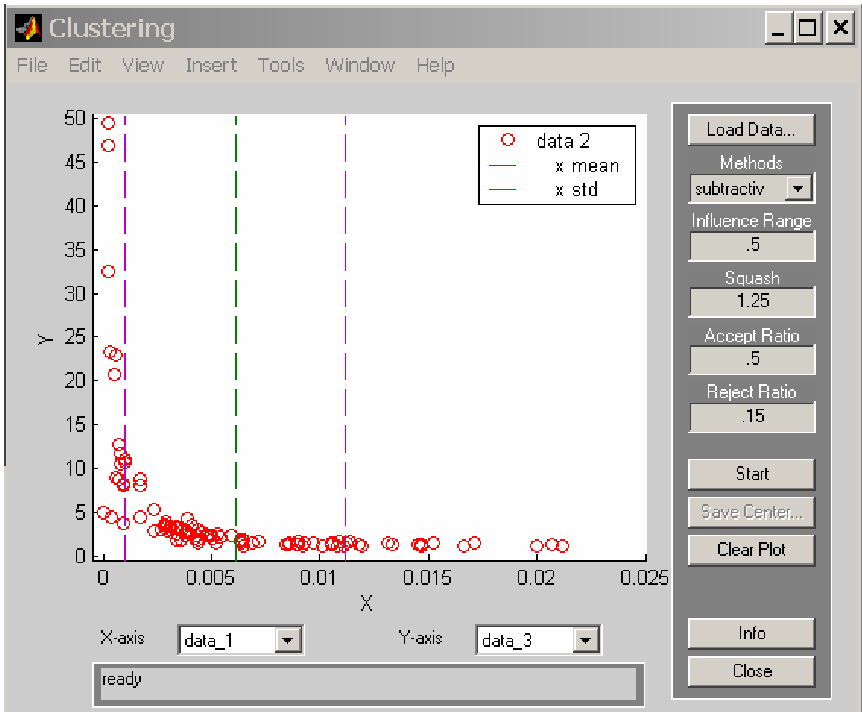

Рис. 4.4.7

Классификация в режиме командной строки

Функция **fcm** осуществляет кластеризацию с использованием алгоритма *Fuzzy C-means* и записывается следующим образом:

[C,U,obj_fcn]=fcm(data, cluster_n, options),

где **data** – матрица данных, каждый столбец которой соответствует одному из признаков, а строка – объекту; **Cluster_n** – число кластеров; **Options** – вектор настроек: **Options(1)** – экспоненциальный вес, **Options(2) -** максимальное число итераций, **Options(3)** – параметр сходимости, **Options(4)** – 1 (если требуется выводить информацию на каждой итерации) или 0 (в противном случае). Если аргумент **Options** опущен, то используются установки по умолчанию.

Функция ***fcm*** возвращает: **C** – матрица центров кластеров (строки соответствуют центрам, столбцы соответствуют координатам); **U** – матрица степеней принадлежности; **obj_fcn** – значение целевой функции, определяющее качество кластеризации.

Функция ***subclust*** осуществляет кластеризацию с использованием алгоритма *Subtractive clustering* и записывается следующим образом:

[C]=subclust(X, radii) или ***[C,S]=subclust(X, radii, xBounds, options)***,

где ***X*** – набор экспериментальных данных (столбцы соответствуют переменным, строки – экспериментальным данным); ***radii*** – вектор, определяющий «области влияния» центров кластеров по каждой входной переменной (если данный аргумент задан как скаляр, то по всем переменным устанавливается один и тот же размер «области влияния»); ***xBounds*** – матрица размерности $2 \times N$, устанавливающая границы областей определения входных переменных (по умолчанию берутся границы изменения признаков); параметр ***options*** имеет следующие значения: ***options(1)*** – Squash Factor (фактор подавления); ***options(2)*** – Accept Ratio – коэффициент принятия; ***options(3)*** - Reject Ratio – коэффициент отклонения; ***options(4)*** – verbose (подробности) – если этот элемент равен 1, то выдается подробная информация о процессе нахождения центров кластеров, по умолчанию – 0.

Функция ***subclust*** возвращает: ***C*** – матрица центров, ***S*** – вектор, который содержит элементы, определяющие диапазоны влияния центров по каждой переменной (для всех центров эти диапазоны одинаковы).

Последовательность действий в режиме командной строки следующая.

1. Загружаем предварительно подготовленные данные (файл Forex.dat) и присваиваем их переменной **X**:
```
>> load Forex.dat;
>> X=Forex.
```
2. Осуществляем классификацию с помощью функции
```
>> [C,U,obj_fcn] = fcm(X,2)
```
или
```
>> [c,s]=subclust(X, radius).
```
3. Вывод графического изображения классификации
```
>> plot(X(:,1), X(:,2), 'o');
hold on;
plot(C(:,1),center(:,2),'o','color','r').
```

ЛИТЕРАТУРА

1. Zimmermann H.-J. Fuzzy Set Theory and its Applications / H.-J. Zimmermann. – Kluwer Academic Publishers, 1997. – 429 p.
2. Заде Л.А. Понятие лингвистической переменной и его применение к принятию приближенных решений / Л.А. Заде. – М.: Мир, 1976. – 168 с.
3. Прикладные нечеткие системы / Под ред. К. Асаи, Д. Ватада, С. Иваи и др. – М.: Мир, 1993. – 368 с.
4. Леоненков А.В. Нечеткое моделирование в среде MATLAB и fuzzyTECH / А.В. Леоненков. – СПБ.: БХВ-Петербург, 2003. – 736 с.
5. Дьяконов В.А., Круглов В.С. Математические пакеты расширения MATLAB. Специальный справочник / В.А. Дьяконов, В.С. Круглов. – СПБ.: Питер, 2001. – 480 с.

СОДЕРЖАНИЕ

Составители: Леденева Татьяна Михайловна
Татаркин Дмитрий Сергеевич
Тарасова Алина Сергеевна

Редактор: Золотарева К.А.