Введение

С целью улучшения оценок адекватности формализации моделей и повышения эффективности функционирования регуляторов и систем автоматического управления (САУ) в условиях неопределенности информации о динамическом поведении сложных объектов управления или о внешней среде в традиционные методы проектирования уже давно включают качественные признаки при описании моделей процессов управления. Обычно к качественным признакам относятся такие понятия, как грубость, чувствительность, устойчивость, адаптация и др. Учет и проверка таких качественных признаков при проектировании регуляторов и САУ позволяют существенно снизить требования к формированию и исследованию неклассических критериев оптимизации. Поиск экстремума по неклассическим критериям оптимизации приводит к необходимости исследования аналогов логико-динамических моделей процессов управления с последующей их лингвистической аппроксимацией.

Анализ результатов имитационного моделирования и практического применения качественных признаков в лингвистической аппроксимации конкретных структур нечетких регуляторов и САУ подтвердил высокую эффективность новой интеллектуальной технологии при создании гибких мобильных систем управления сложными (слабо структурированными) промышленными объектами.

Судьба нечеткой логики (fuzzy logic) как нового научного направления во многом сходна с ее содержанием — необычна, сложна и парадоксальна. Достаточно сказать, что еще в 1989 г. Национальный научный фонд США обсуждал вопрос об исключении fuzzy logic из всех институтских учебников, а годом позже Комитет по контролю над экспортом (COCOM) внес ее в список критически важных оборонных технологий, не подлежащих экспорту потенциальному противнику. Страсти значительно улеглись в настоящее время.

Сегодня элементы нечеткой логики можно найти в десятках промышленных изделий — от систем управления электро-

поездами и боевыми вертолетами до пылесосов и стиральных машин.

В основе нечеткой логики лежит теория нечетких множеств, изложенная в серии работ Лотфи Заде (Lotfi Zadeh) в 1965–1973 гг. Он расширил классическое канторовское понятие множества, допустив, что характеристическая функция (функция принадлежности элемента множеству) может принимать любые значения в интервале [0, 1], а не только значения 0 либо 1. Л. Заде определил также ряд операций над нечеткими множествами и предложил обобщение известных методов логического вывода.

Основная идея состояла в том, что человеческий способ рассуждений, опирающийся на естественный язык, не может быть описан в рамках традиционных математических формализмов. Этим формализмам присуща строгая однозначность интерпретации, а все, что связано с использованием естественного языка, имеет многозначную интерпретацию. С самого начала основная прагматическая цель Заде — создание аппарата, способного моделировать человеческие рассуждения и объяснять человеческие приемы принятия решений в ходе решения различных задач, привлекая в эту область многочисленную армию прикладников. Идеи Заде и его последователей находят применение при создании систем, понимающих тексты на естественном языке, при создании планирующих систем, опирающихся на неполную информацию, при обработке зрительных сигналов, при управлении техническими, социальными и экономическими системами, в системах искусственного интеллекта и робототехнических системах.

Надо сказать, что понятие нечеткого множества вполне согласуется с нашими интуитивными представлениями об окружающем мире. Большая часть используемых нами понятий по своей природе нечетки и размыты, и попытка загнать их в шоры двоичной логики приводит к недопустимым искажениям. Возьмите любой из терминов, которыми мы пользуемся в повседневных дорожных ситуациях: «притормози», «возьми правее», «немного поднажми» и др. Становится понятным кризис традици-

онных экспертных систем и АСУ, оказавшихся неспособными справиться с многообразием и нечеткостью реальных задач.

В ряде случаев нечеткие регуляторы способны обеспечить более высокие показатели качества переходных процессов по сравнению с классическими законами. Использование ПИД-регуляторов в системах подчиненного регулирования позволяет добиться хорошего качества переходных процессов. Однако ПИД-регуляторы не дают ожидаемого результата при изменяющихся во времени параметрах системы, а также при наличии в ее структуре неучтенных нелинейностей. Применение в структуре привода нечеткого регулятора способно придать системе свойство невосприимчивости к стохастическим ограниченным изменениям параметров системы и внешних возмущений.

На сегодняшний день с помощью нечеткой логики реализованы тысячи проектов — от посудомоечных машин, автомобилей и фотокамер с автофокусировкой в потребительском секторе до контроллеров поточного производства, атомных реакторов, аэрокосмических систем и систем военного назначения. Таким образом, изучение нечетких алгоритмов управления становится неотъемлемой задачей современного инженера.

Целью этого методического пособия является разработка и исследование методов нечеткого регулирования, позволяющих повысить показатели регулирования скорости электропривода. Для достижения данной цели решаются следующие задачи: разработка нечеткого логического регулятора с помощью пакета Fuzzy Logic Toolbox; отладка нечеткого логического регулятора; составление базы правил нечеткого логического регулятора; оптимизация показателей регулирования скорости привода.

1. Общие теоретические положения по нечеткой логике

1.1. Определение структуры нечеткого логического регулятора

Современный уровень развития промышленности требует комплексного подхода при разработке систем автоматического регулирования техническими объектами. Это обусловлено, с одной стороны, необходимостью повышения качества управления при минимальных затратах на создание и эксплуатацию систем, с другой стороны — усложнением структуры объекта управления, функций, выполняемых им, и, как следствие, увеличением факторов неопределенности, которые необходимо учитывать для управления объектом.

В настоящее время наиболее широкое применение при решении практических задач получили нечеткие логические регуляторы, позволяющие на основании лингвистической информации, полученной от опытного оператора, управлять сложными, плохо формализуемыми процессами.

Для простейших нечетких алгоритмов рассматриваются функции входа и выхода и не используются функции переходов и операций. Этот тип алгоритмов получил широкое распространение при формализации опыта человека-оператора, управляющего технологическим процессом. Такие алгоритмы называются нечеткими логическими регуляторами.

Структура нечеткого логического регулятора, в котором используются эвристические правила принятия решений, показана на рисунке 1.1. Такие регуляторы используются аналогично традиционным регуляторам с обратной связью.

Рис. 1.1

Структура нечеткого логического регулятора

Определение управляющих воздействий состоит из четырех основных этапов:

а) получение отклонения;

б) преобразование значения отклонения к нечеткому виду, такому как «большой», «средний»;

в) оценка входного значения по заранее сформулированным правилам принятия решения посредством композиционного правила вывода;

г) вычисление детерминированного выхода, необходимого для регулирования процесса.

Описываемый здесь подход значительно расширяет сферу взаимодействия человек — машина посредством формализации нечетких алгоритмов.

Нечеткие логические управляющие устройства характеризуются множеством входных и выходных переменных, определяющих структуру регулятора; множеством лингвистических правил (схемы нечетких рассуждений); нечетким языком. В качестве входных параметров могут быть значения ошибки, интеграл ошибки, значение контролируемых параметров и т. д. В качестве выходной переменной выбирается входное управляющее воздействие управляемой системы.

Для обеспечения необходимых качеств переходных процессов была выбрана структура системы с двумя входными координатами (рис. 1.2), которыми являются ошибка по скорости и ускорение. Выходной координатой разрабатываемой системы

является приращение напряжения управления преобразователем *dU* относительно предыдущего значения.

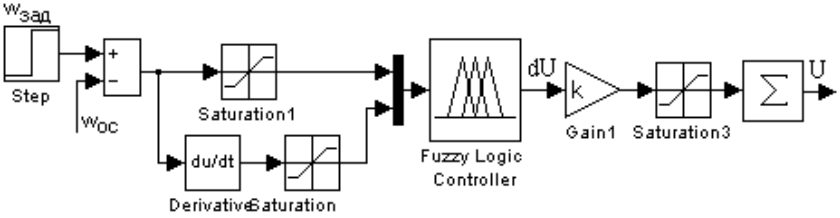

Рис. 1.2

Структура регулятора на основе нечеткой логики

Составление и анализ идеализированной математической модели системы стабилизации проводится с помощью пакета Simulink среды Matlab 6, синтез нечетких регуляторов ведется с помощью пакета Fuzzy Logic Toolbox version 2.1 среды Matlab 6. На выходе нечеткого регулятора формируется величина управляющего воздействия. Выход dU используется для вывода величины приращения управляющего воздействия на текущем пересчете блока.

Значение выхода формируется по следующему алгоритму:

$$U = U_1 + k \cdot dU, \qquad (1.1)$$

где U — управляющее воздействие, В; U_1 – управляющее воздействие на предыдущем такте, В; k — коэффициент усиления приращения управляющего воздействия; dU — приращение управляющего воздействия, В.

Блок Saturation3 предназначен для ограничения величины приращения управляющего воздействия, то есть ограничивает скорость возрастания управляющего напряжения на тиристорном преобразователе.

1.2. Графический интерфейс Fuzzy Logic Toolbox

Собственные программы на основе нечеткой логики анонсировали такие гиганты, как IBM, Oracle и другие. Любопытно, что «взрыв» на рынке программных пакетов на основе нечеткой

логики пришелся примерно на 1995 г. Появилось более сотни программ, использующих нечеткую логику.

Блок Fuzzy Logic Controller (регулятор на основе нечеткой логики) в пакете Simulink имеет один параметр, в котором указывается имя нечеткой системы, смоделированной в редакторе нечеткой системы вывода (FIS Editor) и сохраненной в рабочем пространстве.

Командой (функцией) Fuzzy из режима командной строки запускается основная интерфейсная программа пакета Fuzzy Logic — редактор нечеткой системы вывода. Вид открывающегося при этом окна приведен на рисунке 1.3.

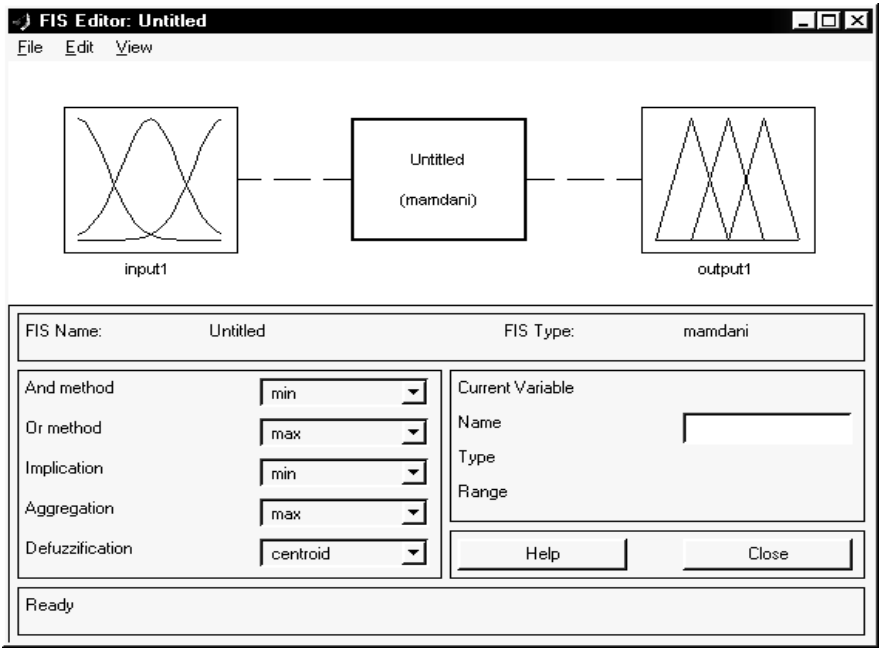

Рис. 1.3

Вид окна редактора нечеткой системы вывода

В меню File выбираем команду New Mamdani FIS (новая система типа Mamdani). Данная система основана на следующем математическом алгоритме.

1. Нечеткость: находятся степени истинности для предпосылок каждого правила: $A_1(x_0)$, $A_2(x_0)$, $B_1(y_0)$, $B_2(y_0)$.

2. Нечеткий вывод: находятся уровни «отсечения» для предпосылок каждого из правил (с использованием операции min):

$$\alpha_1 = A_1(x_0) \wedge B_1(y_0); \quad (1.2)$$

$$\alpha_2 = A_2(x_0) \wedge B_2(y_0), \quad (1.3)$$

где через «∧», как и раньше, обозначена операция логического минимума (min), затем находятся усеченные функции принадлежности:

$$C'_1(z) = (\alpha_1 \wedge C_1(z)); \quad (1.4)$$

$$C'_2(z) = (\alpha_2 \wedge C_2(z)). \quad (1.5)$$

1. Композиция: с использованием операции max (обозначаемой как «∨») производится объединение найденных усеченных функций, что приводит к получению итогового нечеткого подмножества для переменной выхода с функцией принадлежности:

$$\mu_\Sigma(z) = C(z) = C'_1(z) \vee C'_2(z). \quad (1.6)$$

2. Наконец, приведение к четкости (для нахождения z_0) может приводиться несколькими способами (рис. 1.4):

- наименьший максимум (smallest of max, som);
- средний максимум (mean of max, mom);
- наибольший максимум (largest of max, lom);
- центроидный метод — как центр тяжести для кривой $\mu_\Sigma(z)$ (centroid);
- бисекторный (bisector of area).

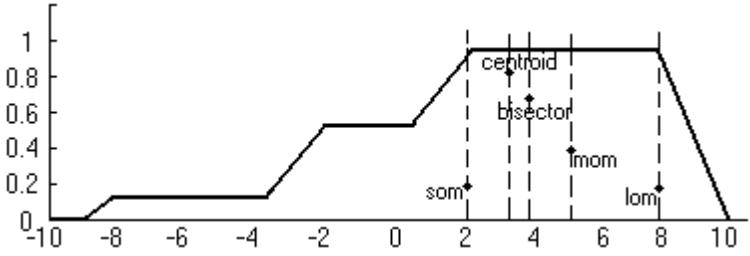

Рис. 1.4
Методы приведения к четкости

Алгоритм Мамдани иллюстрируется на рисунке 1.5 [2].

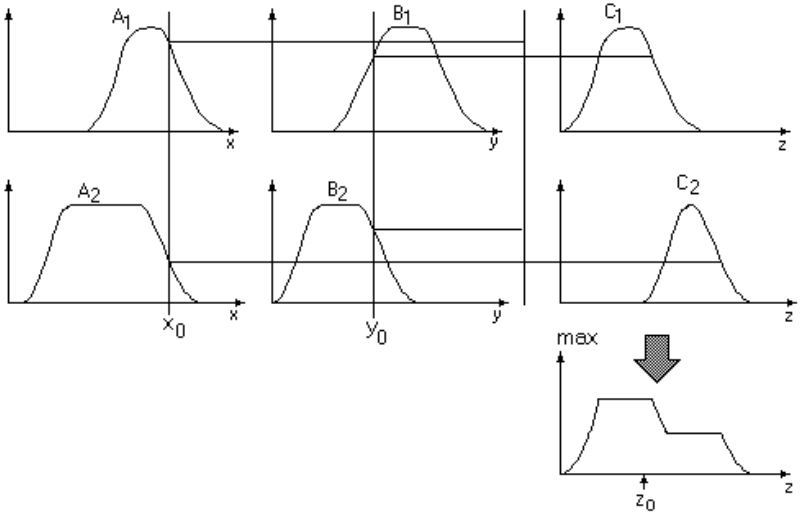

Рис.1.5

Иллюстрации к алгоритму Мамдани

Метод приведения к четкости (defuzzification — дефузификация) оставим по умолчанию: центроидный метод.

Через пункт меню Edit > Add input добавляем в систему второй вход. Делая далее однократный щелчок на блоке input1 и в поле Name, находящемся в правой части редактора, меняем его имя на «Ошибка», завершая ввод имени нажатием клавиши Enter. Аналогичным способом устанавливаем имя «Ускорение» блоку input2 и dU — выходному блоку output1. Присвоим сразу и имя всей системе, например Pusk, выполнив это через пункт меню File > Save to workspace as (Сохранить в рабочем пространстве как...). Через пункт меню File > Save to disk as (Сохранить на диске как...) сохраним систему на жестком диске. Вид окна FIS-редактора после задания структуры системы представлен на рисунке 1.6.

Дважды щелкнув на блоке «Ошибка», откроем окно редактора функций принадлежности (рис. 1.7).

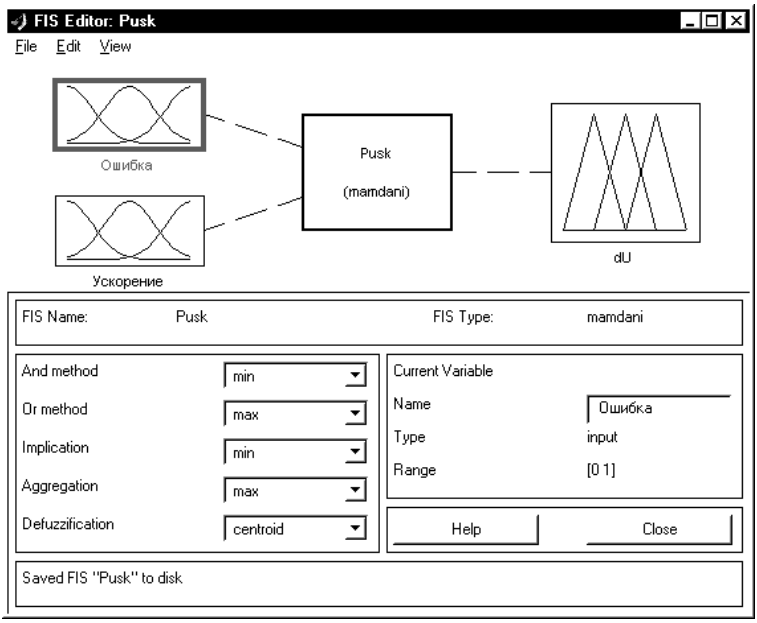

Рис. 1.6

Вид окна FIS-редактора после задания структуры системы

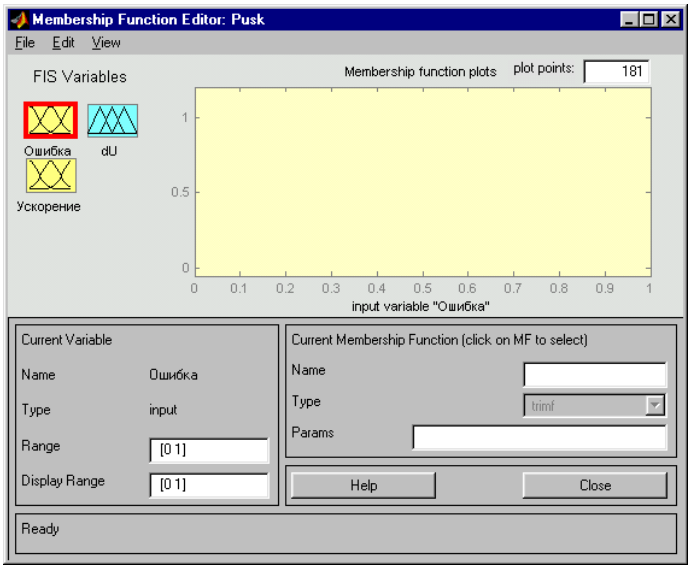

Рис. 1.7

Окно редактора функций принадлежности

Для задания функций принадлежности используются типовые формы приведенных ниже функций ():

— треугольная (trimf);
— трапецеидальная (trapmf);
— гауссова (gaussmf);
— двойная гауссова (gauss2mf);
— обобщенная колоколообразная (gbellmf);
— сигмоидальная (sigmf);
— двойная сигмоидальная (dsigmf);
— произведение двух сигмоидальных функций (psigmf);
— *Z*-функция;
— *S*-функция;
— *Pi*-функция.

Указанные функции принадлежности приведены на рисунке 1.8.

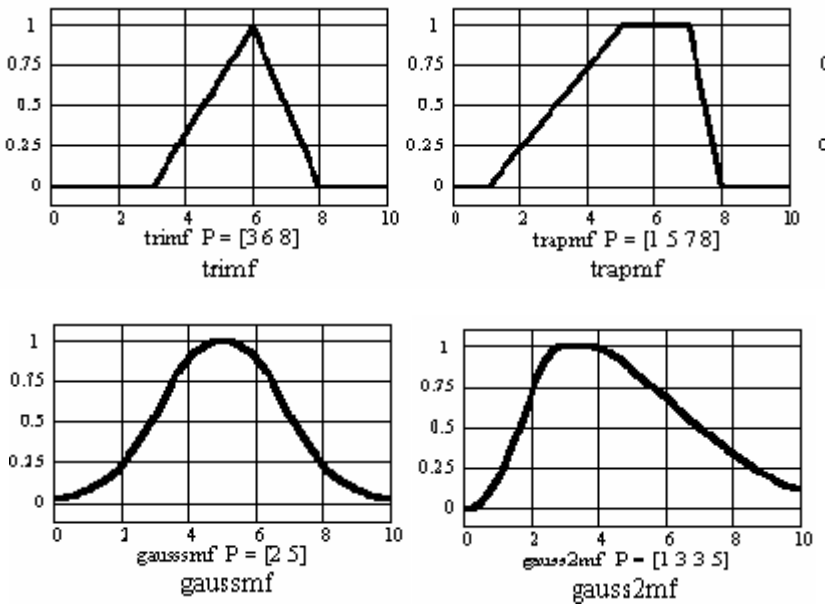

Рис. 1.8. Начало

Типовые функции принадлежности нечетких множеств

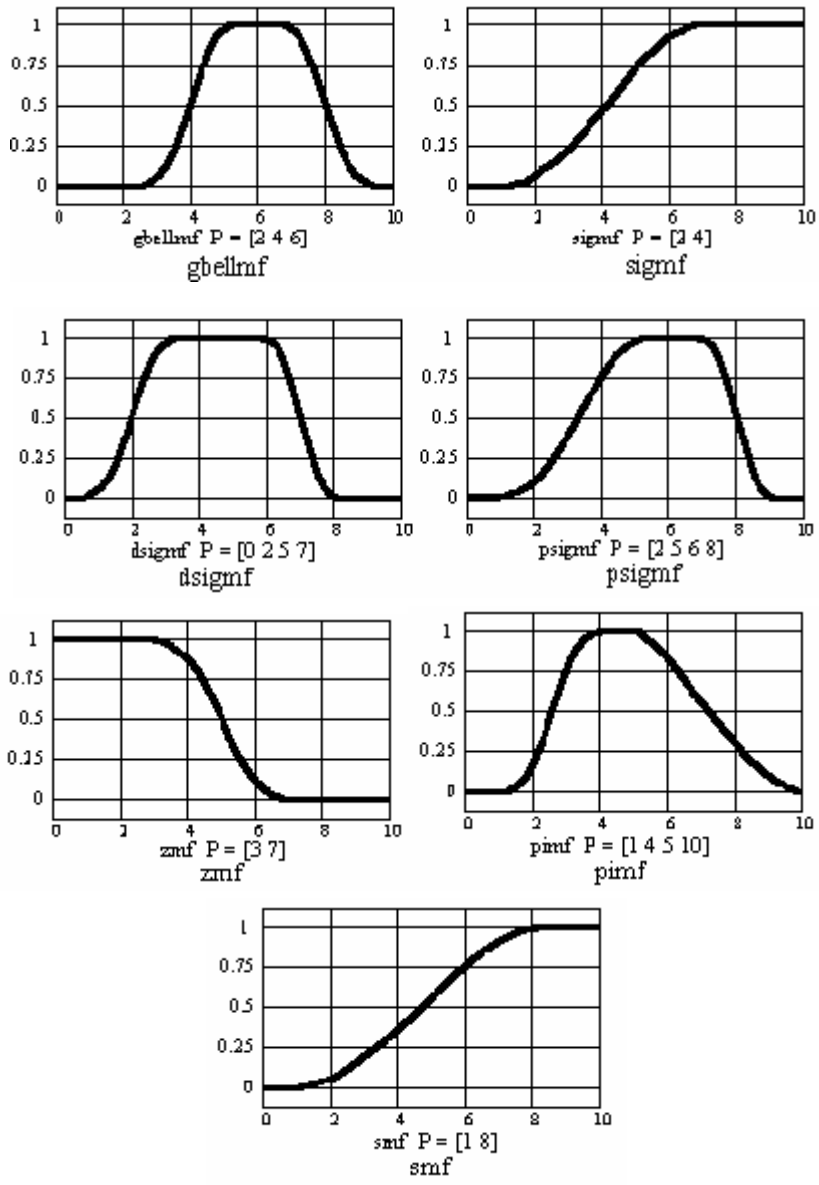

Рис. 1.8. Окончание

Типовые функции принадлежности нечетких множеств (продолжение)

Конкретный вид данных функций определяется значениями параметров, входящих в их аналитические представления, например:

$$\text{trimf}(x,a,b,c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right); \quad (1.7)$$

$$\text{trapmf}(x,a,b,c,d) = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{c-d}\right), 0\right); \quad (1.8)$$

$$\text{gaussmf}(x,\sigma,c) = e^{-\left(\frac{x-c}{\sigma}\right)^2}; \quad (1.9)$$

$$\text{sigmf}(x,a,c) = \frac{1}{1+\exp(-a(x-c))} \quad (1.10)$$

и т. д.

Если по каким-либо причинам вас не устраивает ни одна из встроенных функций принадлежности, можно создать и использовать собственную подходящую функцию.

В поле Range (Диапазон) устанавливается диапазон изменения «Ошибки». В меню Edit данного редактора можно выбрать две команды: Add MFs (добавить функции принадлежности) и Add custom MFs (добавить одну выборочную функцию принадлежности). При выборе Add MFs появится диалоговое окно (рис. 1.9), позволяющее выбрать тип (MF type) и количество (Number of MFs) функций принадлежности.

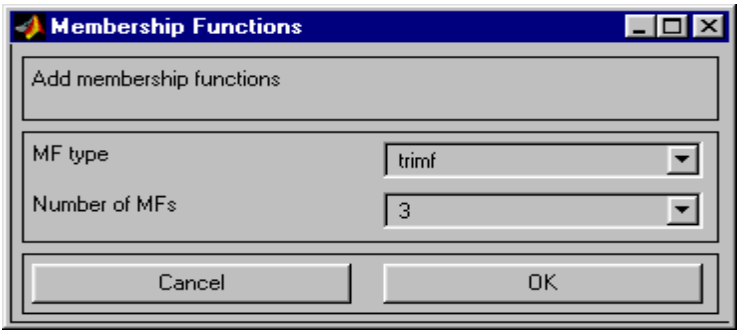

Рис. 1.9

Диалоговое окно задания типа и количества функций принадлежности

Выбранные функции принадлежности отображаются в окне редактора функций принадлежности. С помощью указателя мыши можно выбрать одну из функций принадлежности, после чего с помощью мыши ее можно подвинуть в нужную сторону, изменить вершину, а также увеличить основание треугольника. Более точную установку можно провести, изменяя числовые значения в поле Params (Параметры). В поле Name задается имя кривой (ввод каждого имени завершается нажатием клавиши Enter).

Для конструирования правил необходимо дважды щелкнуть на среднем (белом) блоке, при этом раскроется окно редактора правил (Rule Editor), показанное на рисунке 1.10.

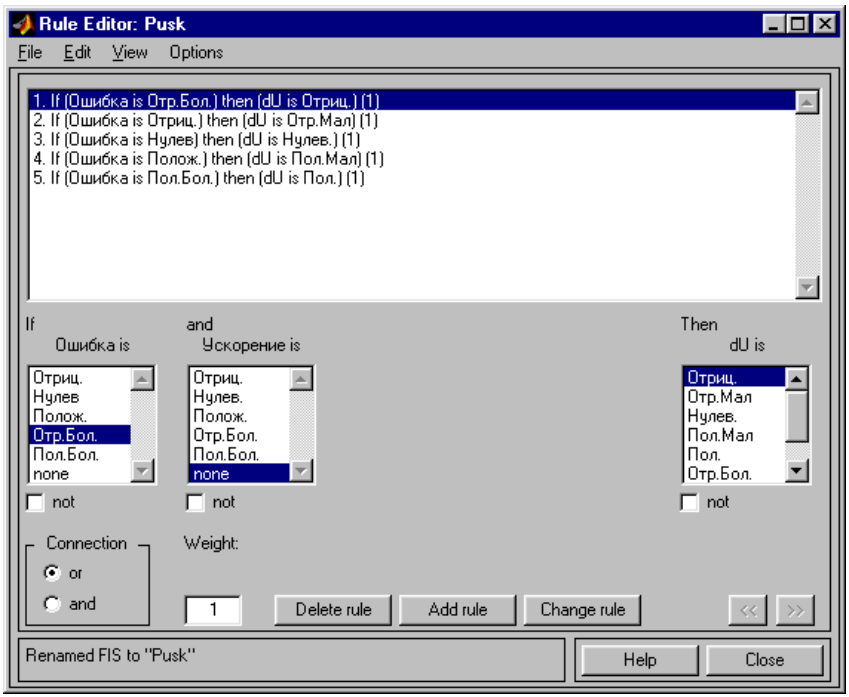

Рис. 1.10

Окно редактора правил

Ввод правил происходит следующим образом:

– выбирается функция принадлежности в столбце «Ошибка» или значение None (Нет);