ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

УПРАВЛЕНИЕ ДИСТАНЦИОННОГО ОБУЧЕНИЯ И ПОВЫШЕНИЯ КВАЛИФИКАЦИИ

Кафедра «Робототехника и мехатроника»

Учебно-методическое пособие

по дисциплине

«Искусственный интеллект в робототехнике и мехатронике»

Авторы

Тугенгольд А.К.,
Юсупов А.Р.,
Изюмов А.И.,
Терехов Д.Ю.

Ростов-на-Дону, 2015

Аннотация

Изложена методика разработки нейро-нечеткой системы аппроксимации нелинейной функции и рассмотрены принципы ее функционирования на базе программного пакета Fuzzy Logic Toolbox.

Авторы

д.т.н., профессор
Тугенгольд Андрей Кириллович

ст.преподаватель
Юсупов Александр Рашидович

аспирант
Изюмов Андрей Игоревич

магистрант
Терехов Дмитрий Юрьевич

Оглавление

Введение

ANFIS является аббревиатурой Adaptive Network – based Fuzzy Inference System. Это основная функция настройки систем нечеткого логического вывода типа Сугэно [1]. Настройка представляет собой итерационную процедуру нахождения параметров системы нечеткого логического вывода, в частности параметров функций принадлежности, которые минимизируют расхождения между результатами логического вывода и экспериментальными данными, т. е. между действительным и желаемым поведениями системы. Экспериментальные данные, по которым настраивается функции принадлежности, представляются в виде обучающей выборки.

В основу функции anfis положен один из методов построения нейро-нечетких систем для аппроксимации функций, предложенный в 1991 году Янгом (Jang). Для идентификации параметров системы нечеткого логического вывода типа Сугэно функция anfis может использовать метод обратного распространения ошибки а также гибридный алгоритм, основанный на комбинации метода обратного распространения ошибки и метода наименьших квадратов. Функция anfis может использовать дополнительный аргумент для проверки модели – тестирующую выборку.

Выполнение лабораторной работы позволит на практике освоить разработку, создание, обучение и использование нейро-нечетких решений для задач моделирования и аппроксимации сложных зависимостей.

Цель работы: освоение основ проектирования нейро-нечёткой системы (ANFIS) аппроксимации нелинейной функции [2] в пакете прикладных программ MatLab.

1.Последовательность выполнения работы

- Сгенерировать обучающую выборку для нейронной сети (НС) по функции.
- Спроектировать нейро-нечеткую систему в ANFIS-редакторе [3].
- Выполнить исследование качества работы ANFIS в зависимости от способа обучения и структуры НС.
- Оптимизировать систему.
- Создать оптимизированную нейро-нечеткую систему в скрипте.

2.Определение исходных данных

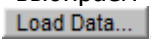
Исходные данные берутся из таблицы 2 в соответствии с вариантом, полученным у преподавателя.

3.Генерация обучающей выборки

Чтобы сгенерировать обучающую выборку в командной строке задаём значения аргумента (x) в виде столбца значений и функцию (y) от него. Далее формируем обучающую выборку (**trndata**) в виде матрицы, где первый столбец – это значения аргумента (x), а второй столбец – это значения функции (y) от аргумента (x) (см. примеры).

4.Проектирование нейро-нечеткой системы в ANFIS-редакторе

Чтобы открыть ANFIS-редактор (рис.1) вводим слово «**anfisedit**» в командной строке.

Для загрузки обучающей выборки в области загрузки данных (**Load data**) выбираем **Type: Training From: worksp.** и нажимаем кнопку 

. В появившемся окне вводим имя нашей выборки (**trndata**) и нажимаем **OK**. Если всё сделано правильно, то в области визуализации появится графическое представление загружаемой выборки.

Чтобы открыть редактор функций принадлежностей выбираем команду **Membership Functions...** во вкладке **Edit** в основном окне ANFIS-редактора.

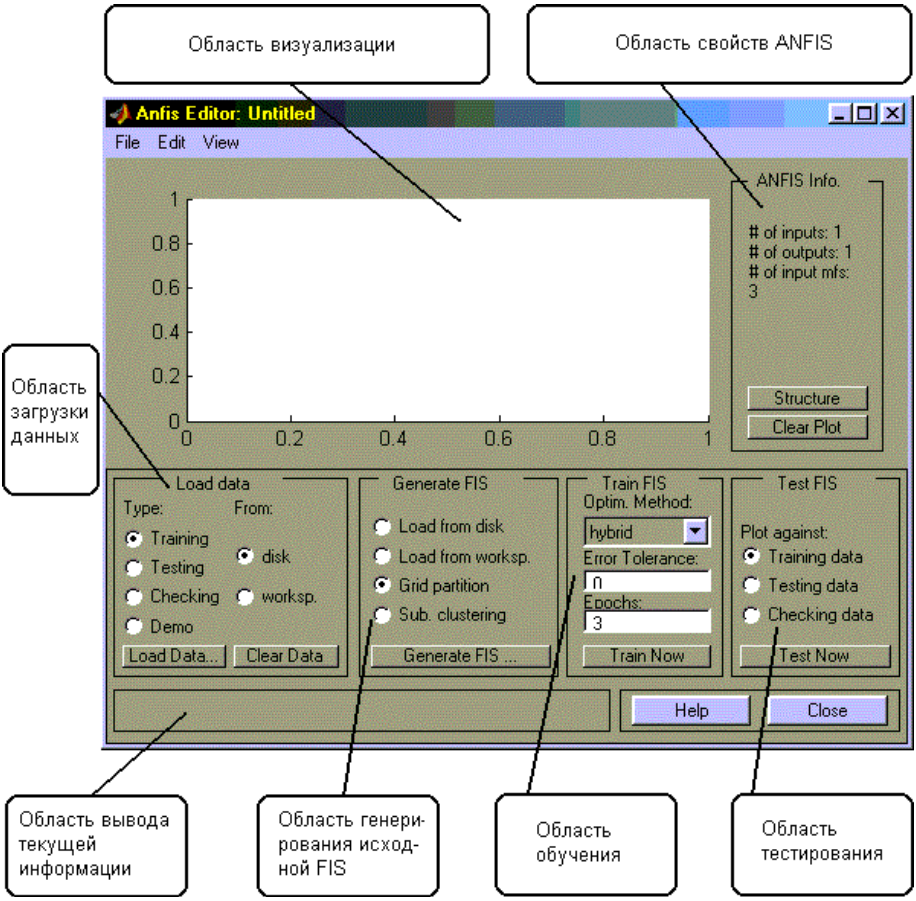


Рисунок 1 – Основное окно ANFIS-редактора

Чтобы генерировать исходную систему нечеткого логического вывода по методу решётки выбираем **Grid partition** в области генерирования исходной FIS (**Generate FIS**) и нажимаем кнопку Generate FIS ... . Далее появится окно ввода параметров метода решётки (рис. 2), в котором указываем количество термов (**Number of MFs**) для каждой входной переменной и тип функций принадлежности (**MF Type**) для входных (**INPUT**) и выходной (**OUTPUT**) переменных.

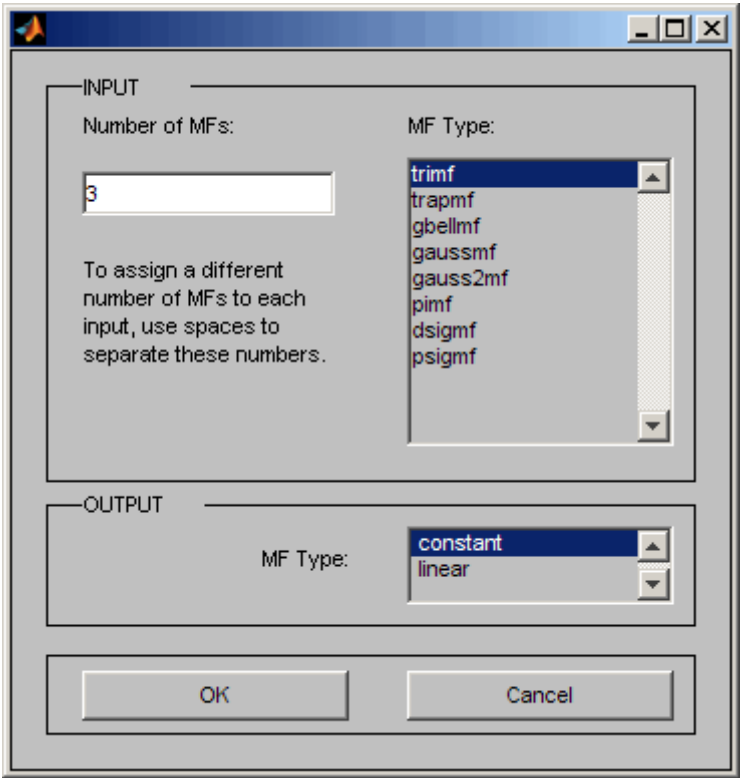


Рисунок 2 – Окно ввода параметров для метода решетки

Выбираем:

- Number of MFs – 3.
- MF Type:
- INPUT – trapmf.
- OUTPUT – linear.

Все изменения будут отображаться в редакторе функций принадлежностей.

Чтобы обучить сеть (**Train FIS**) устанавливаем в области обучения необходимые нам параметры. Здесь расположены меню выбора метода оптимизации (**Optim. method**), поле задания требуемой точности обучения (**Error tolerance**), поле задания количества итераций обучения (**Epochs**) и кнопка **Train Now**, нажатие которой запускает режим обучение. Промежуточные результаты обучения выводятся в область визуализации и в рабочую область MatLab. В ANFIS-редакторе реализованы два ме-

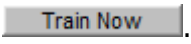
тода обучения:

backpropa – метод обратного распространения ошибки, основанный на идеях метода наискорейшего спуска;

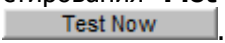
hybrid – гибридный метод, объединяющий метод обратного распространения ошибки с методом наименьших квадратов.

Выбраем:

- Optim. method – backpropa.
- Error tolerance – 0.
- Epochs – 20.

Далее нажимаем кнопку 

.

Чтобы тестировать сеть (**Test FIS**) выбираем в области тестирования **Plot against: Training data**. и нажимаем кнопку 

.

5.Исследование качества работы ANFIS в зависимости от способа обучения и структуры НС

Указанные ранее способ обучения и структура НС не позволяют достичь желаемых критериев качества (средняя ошибка при тестировании менее 0,01). Чтобы уменьшить ошибку необходимо изменить параметры системы. Изменения параметров рекомендуется выполнять в порядке указанном в таблице 1.

Таблица 1 – Последовательность изменения параметров системы

Шаг №	Область	Параметр	Старое значение	Новое значение
1	Train FIS	Epochs	20	100
2	Train FIS	Optim. method	backpropa	hybrid
3	Generate FIS	INPUT MF Type	trapmf	gbellmf
4	Generate FIS	Number of MFs	3	4

После выполнения указанных изменений необходимо переобучить сеть и обратить внимание на величину ошибки обучения. Ошибка существенно снизится, но не исчезнет совсем.

6. Оптимизация системы

Самостоятельно, изменяя параметры метода решётки и структуру НС, настраиваем систему таким образом, чтобы средняя ошибка при тестировании (Average testing error) была менее 0,01.

7. Создание оптимизированной нейро-нечеткой системы в скрипте

Сначала создаём новый скрипт путём нажатия кнопки **New Script** во вкладке **Home** или вводим в командной строке слово «**edit**». Далее вводим в нём программу создания выборки, которую писали в командной строке ранее.

Чтобы генерировать исходную систему нечеткого логического вывода по методу решётки воспользуемся функцией генерирования исходной системы нечеткого логического вывода типа Сугэно из данных без использования кластеризации «**genfis1**» и укажем значения входных аргументов. Аргументы назначаются в следующем порядке **genfis1(data, numMFs, inmftype, outmftype)**. Где:

- **data** – матрица исходных данных, каждая строчка которой является парой “входы – выход”;
- **numMFs** – необязательный аргумент, задающий количество термов для оценки входных переменных. Если количество термов одинаковое для всех переменных, тогда достаточно задать скалярное значение этого аргумента. Значение по умолчанию – 2;
- **inmftype** – необязательный аргумент, задающий типы функций принадлежности термов входных переменных. Значение этого аргумента в виде одной строки символов указывает на то, что все функции принадлежности одного типа. Массив стрингов задает тип функций принадлежностей для каждой входной переменной. По умолчанию используется обобщенная колокообразная функция принадлежности;
- **outmftype** – необязательный аргумент, задающий тип функций принадлежности термов выходных переменных (точнее тип зависимости, связывающей входные и выходную переменные в области действия правила). Допустимые значения: ‘linear’ – линейная и 'constant' – константа. По умолчанию используется линейная зависимость.

Чтобы обучить сеть воспользуемся функцией настройки систем нечеткого логического вывода типа Сугэно «**anfis**» и укажем

значения входных аргументов. Аргументы назначаются в следующем порядке **anfis(trndata, initfis, trnopt, dispopt, chkdata, optmethod)**. Где:

- **trndata** – идентификатор обучающей выборки. Этот входной аргумент является обязательным. Обучающая выборка представляет собой матрицу, каждая строчка которой является парой "входы-выход".
- **initfis** – идентификатор исходной системы нечеткого логического вывода, функции принадлежности которой будут настроены с помощью anfis. Исходная система нечеткого логического вывода должна быть системой типа Сугэно нулевого или первого порядка.
- **trnopt** – вектор параметров настройки:
- trnopt(1) – количество итераций (значение по умолчанию – 10);
- trnopt(2) – допустимая ошибка обучения (значение по умолчанию – 0);
- trnopt(3) – исходная длина шага (значение по умолчанию – 0.01);
- trnopt(4) – коэффициент уменьшения длины шага (значение по умолчанию – 0.9);
- trnopt(5) – коэффициент увеличения длины шага (значение по умолчанию – 1.1);
- **dispopt** – вектор, указывающий какие промежуточные результаты обучения выводить в командное окно MatLab во время настройки:
- dispopt(1) – ANFIS-информация: количество функций принадлежности входных и выходной переменных и т.п.;
- dispopt(2) – ошибка обучения;
- dispopt(3) – длина шага, (выводится в случае его изменения);
- dispopt(4) – окончательный результат.

  По умолчанию значения всех координат равны 1;
- **chkdata** – идентификатор тестирующей выборки.
- **optmethod** – метод оптимизации, используемый для настройки. Допустимые значения: 1 – гибридный алгоритм и 0 – метод обратного распространения ошибки.

Чтобы протестировать сеть построим графики обучающей выборки и результатов моделирования. В этом нам помогут функции:

- **plot(x, y)** – график в линейном масштабе.

- **evalfis(input, fis)** – выполнение нечеткого логического вывода. Где:

- **input** – матрица значений входных переменных, для которых необходимо выполнить нечеткий логический вывод.

- **fis** – идентификатор системы нечеткого логического вывода.

8. Примеры

- Требуется создать обучающую выборку на основе функции $y = \sin(x)$.

Для этого в командной строке вводим:

x=(0:pi/10:3*pi)' – создание последовательности чисел от 0 до 3π с шагом $\pi/10$ и последующим транспонированием (превращением строки чисел в столбец).

y=sin(x) – создание функции $y = \sin(x)$.

trndata=[x y] – создание обучающей выборки.

Далее запускаем программу, нажав на кнопку **Enter**.

- Требуется создать нейро-нечеткую систему в скрипте на основе функции $y = \sin(x)$.

```
x=(0:pi/10:3*pi)';
y=sin(x);
trndata=[x y];
in_fis = genfis1(trndata,5,'gaussmf','linear');
out_fis = anfis(trndata,in_fis,50,[],[],1);
plot(x,y,'o',x,evalfis(x,out_fis),'*');
legend('Training Data','ANFIS Output');
```

Далее запускаем программу, нажав на кнопку **Run** во вкладке **Editor** либо на кнопку **F5** на клавиатуре. Результат будет отображён в виде графиков (рис. 3).

Следует обратить внимание на то, что в качестве примера взята простая функция. На практике же возможностями нейро-нечётких сетей пользуются для решения различных сложных задач, таких как диагностика состояния и прогнозирование остаточной стойкости режущих инструментов в многооперационных станках, моделирование адаптивной системы нейро-нечеткого управления рабочим процессом стрелового крана и т.д.

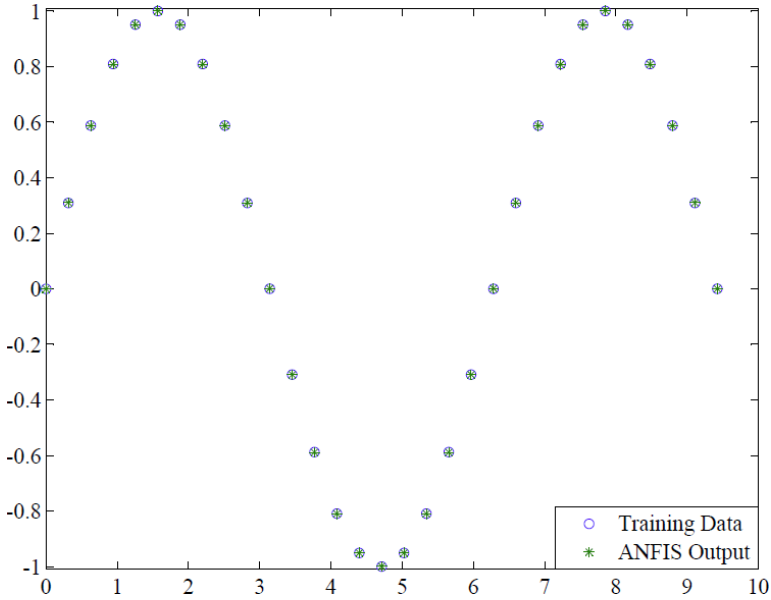


Рисунок 3 – Результаты проектирования

9. Задания

Уравнения функций и параметры их изменений для формирования обучающей выборки приведены в таблице 2.

Таблица 2 – Варианты заданий

№	Вид зависимости	Диапазон изменения
1	y=sin(2.*x)./exp(x./5)	0:0.1:10
2	y=sin(x)+exp(x./15)	0:0.2:20
3	y=cos(3.*x).*exp(x./5)	0:0.1:10
4	y=cos(x)-exp(x./15)	0:0.25:25
5	y=cos(2.*x).*sin(x)	0:0.1:10
6	y=cos(x.^0.9).*sin(x.^1.1)	0:0.15:15
7	y=cos(5.*x.^0.5).*cos(x)	0:0.15:15

8	y=sin(x).^2.*sin(x)	0:0.1:10
9	y=sin(3.*x).^2+sin(x)	0:0.1:10
10	y=exp(2.*sin(x)).*cos(x)	0:0.1:10

Содержание отчета по лабораторной работе

Результаты оформить в виде отчёта. В отчёте представить:

1. Цель работы.
2. Задание.
3. Заключение о влиянии изменения параметров системы в областях генерирования исходной FIS, обучения и визуализации (график) для каждого шага из последовательности изменения параметров (табл. 1) и после оптимизации.
4. Результат выполнения нечеткого логического вывода в графической форме, полученный с помощью оптимизированной нейро-нечеткой системы спроектированной в скрипте и сам скрипт.
5. Результаты проверки качества системы: оценка близости результатов моделирования к данным обучающей выборки (ошибка обучения) для каждого шага из последовательности изменения параметров до и после оптимизации.
6. Выводы.

Вопросы для самоконтроля

1) Что из себя представляет структура ANFIS-сети?
2) Какие алгоритмы применяются при обучении сети ANFIS?
3) Что такое итерация? Как количество итераций влияет на качество обучения?
4) Какие методы обучения реализованы в ANFIS-редакторе?
5) Как подготавливаются данные для обучения сети?
6) Что такое «нечеткие знания»?
7) Дайте определение понятию «функция принадлежности» и укажите типовые формы границ, ее определяющих.

Литература

1. С. Д. Штовба Проектирование нечётких систем средствами MATLAB.
http://matlab.exponenta.ru/fuzzylogic/book1/index.php
2. Искусственный интеллект и интеллектуальное управление

в мехатронике: учеб. пособие / А.К. Тугенгольд. – Ростов н/Д: Издательский центр ДГТУ, 2010. – 146 с.

3. Пакет Fuzzy Logic Toolbox for Matlab : учеб. пособие / В. С. Тарасян. – Екатеринбург : Изд-во УрГУПС, 2013. – 112 с.

http://biblioserver.usurt.ru/