

СЕКЦИЯ 4.
Нейро-сетевые технологии

Председатель:
д. ф.-м. н., профессор М. Н. Рычагов

Оглавление

Агамалов О. Н., Костерев Н. В., Лукаш Н. П. ЭКСПЕРТНАЯ СИСТЕМА ОЦЕНКИ ТЕХНИЧЕСКОГО СОСТОЯНИЯ СИСТЕМЫ ВОЗБУЖДЕНИЯ СИНХРОННОГО ГЕНЕРАТОРА В РЕАЛЬНОМ МАСШТАБЕ ВРЕМЕНИ.....	1234
Аманов К. А., Вощанкин С. В., Смирнов А. Б., Черняк Б. Я. ПРИМЕНЕНИЕ МАТЛАВ ПРИ РАЗРАБОТКЕ СИСТЕМ УПРАВЛЕНИЯ БЕНЗИНОВЫМИ ДВИГАТЕЛЯМИ ВНУТРЕННЕГО СГОРАНИЯ	1255
Артюхин В. В., Соломаха А. А., Горбаченко В. И. ИСПОЛЬЗОВАНИЕ ПАКЕТА NEURAL NETWORK TOOLBOX СРЕДЫ МАТЛАВ ДЛЯ ДИАГНОСТИКИ ГЕПАТИТА У ХИРУРГИЧЕСКИХ БОЛЬНЫХ.....	1262
Злобин В. В. РЕШЕНИЕ РЕСУРСОЕМКИХ ВЫЧИСЛИТЕЛЬНЫХ ЗАДАЧ С ПОМОЩЬЮ НЕЙРОСЕТЕЙ	1266
Зюзев А. М., Костылев А. В. СИСТЕМА ДИАГНОСТИКИ ШТАНГОВОЙ ГЛУБИННО-НАСОСНОЙ УСТАНОВКИ НА ОСНОВЕ НЕЙРОННОЙ СЕТИ	1273
Локтионов А. А., Аргынова А. Х., Лось В. Л., Токарский Э. А. РАССТОЯНИЕ МЕЖДУ ОБЪЕКТАМИ ПРИ ПРОГНОЗЕ ПОЛЕЗНЫХ ИСКОПАЕМЫХ.....	1288
Мещеряков В. А. ИДЕНТИФИКАЦИЯ СТРОИТЕЛЬНЫХ МАШИН КАК НЕЛИНЕЙНЫХ ДИНАМИЧЕСКИХ СИСТЕМ НА ОСНОВЕ НЕЙРОСЕТЕВЫХ ТЕХНОЛОГИЙ.....	1300
Хрящёв В. В., Соколенко Е. А., Приоров А. Л. СРАВНЕНИЕ ЭФФЕКТИВНОСТИ РЕАЛИЗАЦИИ АЛГОРИТМОВ ОБУЧЕНИЯ НЕЙРОСЕТИ В ЗАДАЧЕ ВОССТАНОВЛЕНИЯ ИЗОБРАЖЕНИЙ	1309
Штовба С. Д., Панкевич О. Д. ПРОЕКТИРОВАНИЕ НЕЧЕТКИХ КЛАССИФИКАТОРОВ В СИСТЕМЕ МАТЛАВ	1320
Юзбашев Д. А. НЕЙРОННЫЕ СЕТИ ТРЕТЬЕГО ПОКОЛЕНИЯ. РАСПОЗНАВАНИЕ ОБРАЗОВ И СКРЫТЫХ ЗАВИСИМОСТЕЙ В ПРОИЗВОЛЬНОМ СИГНАЛЕ С ПОМОЩЬЮ ИМПУЛЬСНЫХ НЕЙРОСЕТЕЙ С ИСПОЛЬЗОВАНИЕМ СРЕДЫ МАТЛАВ.....	1336

УДК 621.316.722

ЭКСПЕРТНАЯ СИСТЕМА ОЦЕНКИ ТЕХНИЧЕСКОГО СОСТОЯНИЯ СИСТЕМЫ ВОЗБУЖДЕНИЯ СИНХРОННОГО ГЕНЕРАТОРА В РЕАЛЬНОМ МАСШТАБЕ ВРЕМЕНИ

Агамалов О. Н.,

*Южноукраинская АЭС, Южноукраинск, Украина,
e-mail: aon@ukrsat.net.ua*

Костерев Н. В.,

*Национальный технический университет Украины
«Киевский политехнический институт», Киев, Украина,
e-mail: kpi_power@zeos.net*

Лукаш Н. П.

*Национальный технический университет Украины
«Киевский политехнический институт», Киев, Украина,
e-mail: kpi_power@zeos.net*

1. Введение

Техническая диагностика электрооборудования, как и любой другой технической системы, основана на следующих основных положениях [1]:

- а) последовательные и систематические измерения параметров;
- б) выявление изменений параметров и сравнение их с исходными.

Диагностирование электрооборудования возможно на основании априорных сведений, полученных при проведении его испытаний (режим off-line) или на основании информации об изменении параметров в процессе эксплуатации (режим on-line). Существуют следующие методы диагностики нарушений в процессе эксплуатации [2]:

1. Статистический — основан на известных вероятностных соотношениях между неисправностью (ее симптомами) и наблюдаемыми изменениями параметров, используя оценки функций правдоподобия методами байесовского анализа.

2. Детерминистический — основан на анализе схемы технологического процесса (диагностируемого объекта) и выявлении тех точек, в которых необходимо проверить наличие симптомов нарушений.

3. Распознавание последовательности симптомов — метод, основанный на сравнении реальной последовательности симптомов (признаков) нарушения работы с эталонными, хранимыми в базе знаний.

4. Создание полных математических моделей диагностируемых объектов (процессов) — наблюдаемое состояние относят к наиболее близкой математической модели.

Все множество режимов работы электрооборудования может быть представлено как объединение подмножеств установившихся и переходных режимов. В установившихся режимах работы могут быть определены статические характеристики электрооборудования. В переходных режимах могут быть определены динамические характеристики оборудования, коэффициенты передачи и постоянные времени, их зависимости от частоты. Анализ переходных режимов работы позволяет наиболее точно оценить техническое состояние электрооборудования и спрогнозировать его изменение в будущем. Изменения параметров электрооборудования в переходных режимах работы характеризуются неопределенностью и нелинейностью. Неопределенность определения параметров в переходных режимах работы электрооборудования вызвана следующими факторами:

- низкая точность измерения штатными приборами контроля в переходных режимах работы;
- отказы каналов измерения или датчиков при выходе параметра за допустимый диапазон измерения в переходном режиме работы;
- запаздывание при передаче информации в каналах измерения.

2. Постановка задачи

Существующим методам диагностики оборудования в процессе эксплуатации присущи следующие недостатки. С помощью статистического метода невозможно оценить техническое состояние оборудования в переходных режимах [2]. В условиях неоднородной информации, характеризующей переходный режим работы, данный метод просто не работает. Детерминистический метод не позволяет контролировать весь объект, адаптировать и обучать систему контроля при изменении внешних условий и режимов работы оборудования. Метод распознавания последовательности симптомов позволяет легко расширять базу последовательностей признаков, однако для его реализации необходимо хранить однотипные эталонные последовательности признаков и трудно представлять нецифровую (лингвистическую) информацию о происходящих процессах. Построение полных математических моделей для сложных, параллельных процессов, происходящих в электрооборудовании, пока практически невозможно.

В докладе рассматривается методика оценки технического состояния электрооборудования, на примере синхронного генератора с бесщеточной системой возбуждения, реализованная в двухуровневой экспертной системе (ЭС) реального времени. Первый уровень ЭС представляет собой классификатор переходных режимов работы системы возбуждения, анализирующий данные, полученные от микропроцессорного регистратора в реальном масштабе времени. Его задачей является определение принадлежности зарегистрированного переходного режима к классам нормальных или аномальных. Нормальным переходный режим работы определяется в

том случае, если его причиной было возмущение в энергосистеме (КЗ, синхронные качания, неполнофазные режимы и т. д.) или изменения в работе системы автоматического регулирования частоты вращения (АРЧВ) турбины, внешние по отношению к контролируемому объекту. Анормальный переходный режим работы определяется в том случае, если его причиной было возмущение в работе системы возбуждения синхронного генератора, вызванное ее неисправностью. В случае, если текущий переходный режим работы определен как анормальный, с помощью второго уровня ЭС — идентификатора, определяется конкретная неисправная подсистема (элемент) оборудования системы возбуждения. Поиск неисправных элементов основан на сравнении реальных выходов с выходами нечетких нелинейных авторегрессионных моделей. Модели создаются для исправного технического состояния оборудования и отражают изменения входных и выходных параметров основных устройств системы возбуждения во всех возможных режимах работы. Выход модели $y_m(k)$ в k -й момент времени при исправном техническом состоянии оборудования должен соответствовать реальному выходу $y(k)$ с минимальной ошибкой:

$$\delta = \sqrt{\frac{1}{N} \sum_{k=0}^{N-1} (y(k) - y_m(k))^2} \rightarrow \min, \quad (1)$$

где N — количество точек анализируемой выборки переходного процесса.

3. Алгоритм работы и настройка параметров экспертной системы

Режимы работы системы возбуждения синхронного генератора могут быть разбиты на классы, исходя из соотношений между изменениями уставки АРВ, параметрами статора и ротора синхронной машины:

1. установившийся (стационарный) режим;
2. пусковой режим (начальное возбуждение);
3. останов (гашение поля);
4. форсировочный режим;
5. расфорсировочный режим;
6. синхронные качания;
7. асинхронный ход.

Режимы 1–5 могут быть нормальными, т. е. возникающими в результате реакции системы возбуждения на внешние возмущения, или анормальными, при возникновении неисправности в оборудовании системы возбуждения. Режимы 6, 7 могут быть представлены как наложения режимов 4, 5 до возникновения установившегося режима или гашения поля возбуждения. Поэтому можно определить базовую классификацию режимов работы системы возбуждения (из которой могут быть получены все остальные), используемую в работе ЭС:

1. установившийся режим (нормальный и анормальный);
2. форсировочный режим (нормальный и анормальный);

3. расфорсировочный режим (нормальный и аномальный).

Существенными нечеткими признаками аномальных режимов работы системы возбуждения, положенными в основу базы нечетких правил классификатора, являются:

- в аномальном установившемся режиме система возбуждения не реагирует на изменение параметров статора генератора;
- в аномальном форсировочном режиме параметры статора и ротора синхронного генератора увеличиваются относительно значений начального установившегося режима;
- в аномальном расфорсировочном режиме параметры статора и ротора синхронного генератора уменьшаются относительно значений начального установившегося режима.

Предполагаем, что начальный установившийся режим работы системы возбуждения и синхронного генератора является нормальным и соответствует исправному состоянию оборудования системы возбуждения. Поэтому классификация регистрируемого переходного режима осуществляется на основании анализа отклонений параметров синхронного генератора и системы возбуждения относительно значений начального установившегося режима. Описание всех возможных режимов работы электрооборудования требует создания объемной базы правил. Например, для объекта, имеющего множество входов n_u и множество выходов n_y (ММО — many input, many output), общее количество правил, необходимых для его описания

$$L = \prod_{i=1}^{n_u+n_y} N_i, \quad (2)$$

где N_i — количество нечетких термов для описания i -й переменной.

Упрощение базы нечетких правил может быть достигнуто, если на переменные посылок наложить некоторые ограничения. В соответствии с (2) ограничения могут быть наложены на размерность вектора входов, выходов или количество нечетких термов для описания i -ой переменной посылок. При ограничении размерности векторов входа и выхода объекта в базе нечетких правил могут быть потеряны существенные закономерности, отражающие его функционирование и, соответственно, техническое состояние. Уменьшение количества нечетких термов также сказывается на качестве моделирования объекта во всех возможных в эксплуатации режимах работы. Однако при определенных условиях можно задать минимально необходимое количество нечетких термов так, чтобы база правил моделировала вектор выхода при исправном техническом состоянии объекта.

Рассмотрим, каким образом формируется анализируемая выборка переходного режима работы электрооборудования. Начальные условия, т. е. значения векторов входа \mathbf{u}_0 и выхода \mathbf{y}_0 объекта до возникновения переходного режима работы и момент его возникновения определяются в соответствии с критериями, предложенными в [3]. Если задать допустимую

погрешность измерений и представить значения \mathbf{u} и \mathbf{y} в виде симметричных гауссовых функций принадлежности с нормальным законом распределения, центр которых определяется значениями в начальном установившемся режиме \mathbf{u}_0 и \mathbf{y}_0

$$\begin{aligned}\mu_u(u) &= \exp\left(-\frac{(\mathbf{u} - \mathbf{u}_0)^2}{2\sigma_u^2}\right); \\ \mu_y(y) &= \exp\left(-\frac{(\mathbf{y} - \mathbf{y}_0)^2}{2\sigma_y^2}\right),\end{aligned}\quad (3)$$

то критерий возникновения переходного режима работы электрооборудования может быть формализован в нечетком виде

$$\mu_{\text{notN}}(\mathbf{u}, \mathbf{y}) = 1. \quad (4)$$

Логическое условие (4) выполняется при выходе за границу нечеткого гауссового терма N (normal) векторов входа \mathbf{u} или выхода \mathbf{y} . Область терма N определяется экспертами на основании собственного опыта и технических характеристик оборудования как отклонения $\pm d\mathbf{u}(d\mathbf{y})$. Логическое условие, определяющее окончание переходного процесса может быть сформулировано в виде

$$\mu_N(\mathbf{u}, B) = M. \quad (5)$$

с заданным значением $M \in (0, 1)$. Критерии (4) и (5) определяют выборку переходного процесса, на которой оценивается техническое состояние электрооборудования. Степень увеличения или уменьшения координат векторов входа и выхода объекта может быть определена, если ввести соответствующие нечеткие термы L — малого и H — большого значений, определяемые относительно значений терма нормальных значений N. Таким образом, в рассматриваемых точках переходного режима работы параметры посылок базы нечетких правил представляются тремя нечеткими термами относительно значений начального установившегося режима:

- L (low) — если параметр снижается относительно значения в начальном установившемся режиме;
- N (normal) — если отклонение параметра не выходит за определенный диапазон $\pm d\mathbf{u}(d\mathbf{y})$ относительно значения в начальном установившемся режиме;
- H (high) — если параметр повышается относительно значения в начальном установившемся режиме;

Если созданная модель не может описать какой-либо из режимов работы оборудования при его исправном техническом состоянии, количество термов (и соответственно количество правил) параметров посылок может быть увеличено. Этим достигается последовательная адаптация базы нечетких правил ко всем возможным в эксплуатации режимам работы оборудования.

Переходный режим работы электрооборудования может возникнуть в случайный момент времени при произвольных начальных значениях вектора входов \mathbf{u} и выходов \mathbf{y} . Поэтому при описании посылок нечетких правил базы знаний предлагается использовать символьную нечеткую переменную [4], расширяющую существующее определение лингвистической переменной. Отличие заключается в том, что границы лингвистических термов переменных не задаются заранее жестко численными значениями, а подстраиваются к значениям текущего установившегося режима работы электрооборудования и определяются путем задания экспертом допустимого отклонения $\pm du(dy)$ для терма нормальных значений N. Отклонения $\pm du(dy)$ в символьном виде также задают области функций принадлежности термов L и H при изменении их от 0 до 1. Выбираются гладкие, дифференцируемые функции: симметричного гауссового типа для термов N, L, H или Z-образной формы для термов L и S-образной формы для термов H [5]. Определение символьной нечеткой переменной для i -го входа объекта u_i показано на рис. 1.

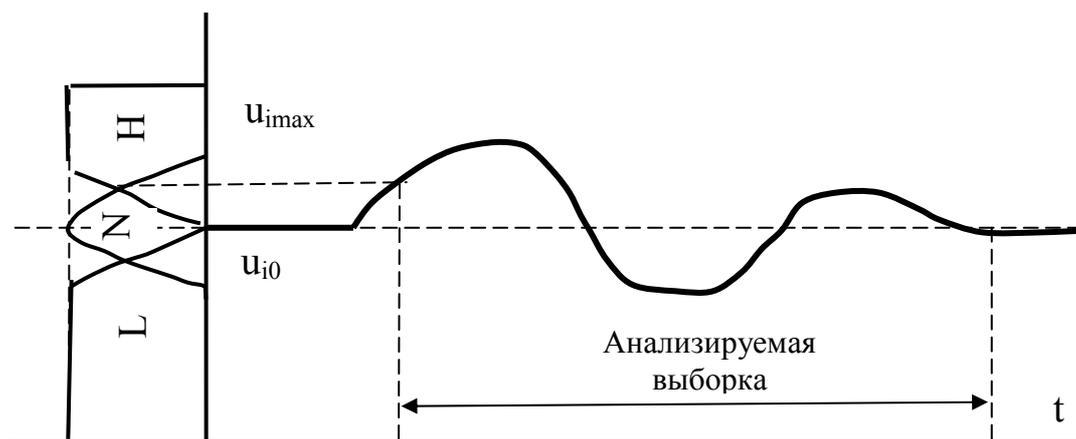


Рис. 1. Определение для i -го входа объекта u_i символьной нечеткой переменной.

Таким образом, при возникновении нового переходного режима работы, значения символьной нечеткой переменной каждый раз подстраиваются (адаптируются) к значениям предшествующего установившегося режима работы, что позволяет описать лингвистическую переменную небольшим количеством нечетких термов.

Для классификации переходного режима работы системы возбуждения необходимо оценить изменения ее входных параметров — напряжения U_s , тока I_s и частоты f статора синхронного генератора и выходных параметров — напряжения U_d и тока I_d ротора. Основным параметром, на который реагирует система возбуждения, является напряжение статора U_s , а режим работы системы возбуждения в целом задается выходом АРВ U_A . Пуск ЭС выполняется в соответствии с логическим условием (4) $\mu_{\text{notN}}(U_A, U_s) = 1$, характеризующим выход за терм N (not N) символьных

нечетких переменных U_d или U_s . Значения центров термов N символьных нечетких переменных определяются за 200 мс до возникновения переходного режима (условие (4)). Информация о последующих значениях параметров системы возбуждения и синхронного генератора образует оперативную базу данных классификатора для анализируемого переходного режима работы. Условие окончания анализа переходного режима работы в соответствии с (5) определяется как $\mu_N(U_d, U_s) = 1$. Для учета динамических свойств бесщеточного возбудителя и обмотки возбуждения синхронного генератора их параметры оцениваются для различных точек переходного режима работы. ЭС, определив возникновение переходного режима работы по условию (4) в момент времени t_i , рассматривает значение напряжения выхода тиристорного преобразователя u_f также в момент времени t_i (инерционностью тиристорного преобразователя порядка нескольких мкс пренебрегаем). Для учета инерционности возбудителя с постоянной времени τ_e , функция принадлежности символьной нечеткой переменной напряжения ротора U_d определяется для момента времени $t_{i+\tau_e}$. Значения функций принадлежности для символьной нечеткой переменной тока ротора I_d оцениваются для момента времени $t_{i+T_d'}$, где T_d' — переходная постоянная времени обмотки ротора.

Прототип ЭС реализован на базе пакета нечеткой логики Fuzzy Logic Toolbox системы MATLAB. Механизм логического вывода классификатора основан на алгоритме Сугэно (Sugeno) 1-го порядка, в соответствии с которым множеству значений входных переменных определено значение интегрального параметра MODE, характеризующего режим, табл. 1.

Таблица 1

U_s	I_s	f	U_d	I_d	MODE	Режим работы и состояние оборудования
N	N	N	N	N	$k_{11}U_s+k_{12}I_s+k_{13}f+k_{14}U_d+k_{15}I_d+m_1$	Нормальный установившийся режим (NS). Оборудование исправно.
not N	not N	not N	N	N	$k_{21}U_s+k_{22}I_s+k_{23}f+k_{24}U_d+k_{25}I_d+m_2$	Анормальный установившийся режим (ANS). Оборудование неисправно.
not H	not L	not L	H	H	$k_{31}U_s+k_{32}I_s+k_{33}f+k_{34}U_d+k_{35}I_d+m_3$	Нормальный форсировочный режим (NF). Оборудование исправно
not L	not L	not H	not L	not L	$k_{41}U_s+k_{42}I_s+k_{43}f+k_{44}U_d+k_{45}I_d+m_4$	Анормальный форсировочный режим (ANF). Оборудование неисправно
not L	not H	not H	L	L	$k_{51}U_s+k_{52}I_s+k_{53}f+k_{54}U_d+k_{55}I_d+m_5$	Нормальный расфорсировочный режим (NDF). Оборудование исправно
not H	not H	not L	not H	not H	$k_{61}U_s+k_{62}I_s+k_{63}f+k_{64}U_d+k_{65}I_d+m_6$	Анормальный расфорсировочный режим (ANDF). Оборудование неисправно

Коэффициенты $k_{11} \dots k_{65}$ и постоянные $m_1 \dots m_6$ настраиваются таким образом, чтобы абсолютное значение выходного параметра $MODE$ было меньше единицы для всех нормальных режимов работы и равно или более единицы для аномальных режимов работы. Значение выходной переменной $MODE$ может быть интерпретировано в системе координат пространства 5 измерений, определяемых входными переменными базы правил. Начальная точка отсчета в таком пространстве определяется значениями входных переменных базы правил в начальном установившемся режиме работы. Геометрической интерпретацией $MODE$ является гиперсфера с единичным радиусом, отделяющая множество комбинаций входных переменных в нормальных и аномальных переходных процессах. Таким образом, определение аномального переходного режима работы формализуется следующим неравенством:

$$|MODE| \geq 1. \quad (6)$$

Настройка коэффициентов $k_{11} \dots k_{65}$ и постоянных $m_1 \dots m_6$ осуществляется на выборках реальных переходных режимов работы, зарегистрированных с помощью микропроцессорного регистратора. База нечетких правил классификатора (табл. 1) реализована в виде адаптивной нейро-нечеткой сети ANFIS (сокращение от англ., Adaptive Neuro-Fuzzy Inference System) [5, 6].

Если классификатор ЭС определяет текущий переходный режим работы системы возбуждения как аномальный, то принятие решения передается на 2-ой уровень ЭС — идентификатор, который определяет отказавший элемент системы возбуждения. Определение неисправного элемента основано на построении аппроксимирующих моделей для основных устройств системы возбуждения — автоматического регулятора возбуждения (АРВ), тиристорного преобразователя и его системы управления тиристорами (ТП с СУТ), бесщеточного возбудителя (БВД). Настройка моделей производится на выборках типовых переходных режимов работы — короткие замыкания, качания, форсировка-расфорсировка возбуждения и т. д., при исправном техническом состоянии оборудования.

Для отражения динамических свойств оборудования системы возбуждения используется нелинейная авторегрессионная модель с внешним входом (NARX) [7], которая устанавливает нелинейное преобразование между предыдущими значениями входов-выходов рассматриваемой подсистемы (элемента) системы возбуждения и будущим, предсказываемым значением выхода при исправном техническом состоянии. Дискретная NARX-модель объекта описывается уравнением вида

$$y(k) = F \{ y(k-1), \dots, y(k-k_y), u(k-k_d), \dots, u(k-k_u) \}, \quad (7)$$

где k_y — максимальный шаг учитываемых предшествующих значений вектора выхода системы, k_u — максимальный шаг учитываемых предшествующих значений вектора входа системы, k_d — шаг запаздывания измене-

ния вектора выхода по отношению к изменению вектора входа, определяемый свойствами объекта, F — нечеткое преобразование.

NARX-модель показана на рис. 2.

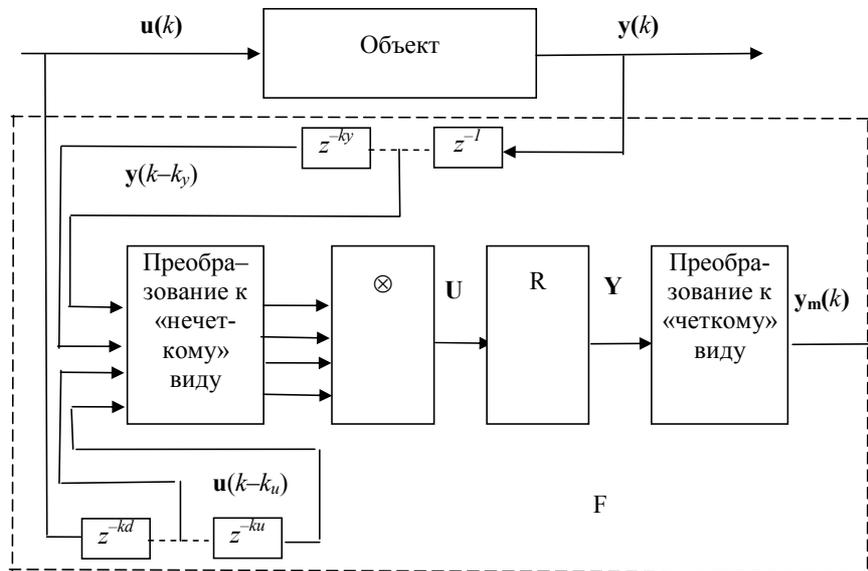


Рис. 2. NARX-модель объекта.

На рис. 2: \otimes — обозначение нечеткого декартового произведения, определяемого на функциях принадлежности векторов входа $\mathbf{u}(k-i)$, $i = k_d, \dots, k_u$ и выхода $\mathbf{y}(k-j)$, $j = 1, \dots, k_y$ в предыдущие моменты времени в виде:

$$\begin{aligned}
 U &= \mu_{u(k-k_d) \times \dots \times u(k-k_u) \times y(k-1) \times \dots \times y(k-k_y)}(\mathbf{u}, \mathbf{y}) = \\
 &= \otimes \left\{ \mu_{u(k-k_d)}(\mathbf{u}), \dots, \mu_{u(k-k_u)}(\mathbf{u}), \mu_{y(k-1)}(\mathbf{y}), \dots, \mu_{y(k-k_y)}(\mathbf{y}) \right\} = \\
 &= \mu_{u(k-k_d)}(\mathbf{u}) \times \dots \times \mu_{u(k-k_u)}(\mathbf{u}) \times \mu_{y(k-1)}(\mathbf{y}) \times \dots \times \mu_{y(k-k_y)}(\mathbf{y}),
 \end{aligned} \tag{8}$$

\mathbf{R} — обозначение композиции, определяющей нечеткое значение выхода

$$\mathbf{Y} = \mathbf{R} \left\{ \mu_{u(k-k_d) \times \dots \times u(k-k_u) \times y(k-1) \times \dots \times y(k-k_y)}(\mathbf{u}, \mathbf{y}) \right\}. \tag{9}$$

Для определения выхода объекта используется алгоритм нечеткого вывода Сугено в векторной форме:

$$\begin{aligned}
 &\text{ЕСЛИ} \left[\begin{array}{l} (\mathbf{u}(k-k_d) \text{ есть } A_{ik_d}) \text{ И } \dots \text{ И } (\mathbf{u}(k-k_u) \text{ есть } A_{ik_u}) \text{ И} \\ (\mathbf{y}(k-1) \text{ есть } B_{i1}) \text{ И } \dots \text{ И } (\mathbf{y}(k-k_y) \text{ есть } B_{ik_y}) \end{array} \right] \text{ТО} \\
 &y_k = a_{i1}(\mathbf{u}(k-k_d)) + \dots + a_{ik_u}(\mathbf{u}(k-k_u)) + b_{i1}(\mathbf{y}(k-1)) + \dots + b_{ik_y}(\mathbf{y}(k-k_y)) + c_i, \tag{10} \\
 &i = 1 \dots L,
 \end{aligned}$$

где $(A_{ik_d} \dots A_{ik_u})$ и $(B_{i1} \dots B_{ik_y})$ — обозначение нечетких множеств и соответствующих им функций принадлежности для посылок i -го правила, в котором

учитываются предыдущие $(k_u - k_d)$ значений вектора входов и $(k_y - 1)$ значений вектора выходов, L — рассматриваемое множество режимов работы объекта, определяющее количество правил, $(a_{ikd} \dots a_{iku})$, $(b_{il} \dots b_{iky})$, c_i — параметры вывода нечеткой модели.

Значения вектора входа $\mathbf{u}(k - k_d), \dots, \mathbf{u}(k - k_u)$ и выхода $\mathbf{y}(k - 1), \dots, \mathbf{y}(k - k_y)$ образуют вектор прогнозирования

$$\mathbf{Z} = [\mathbf{u}(k - k_d) \dots \mathbf{u}(k - k_u)] \cup [\mathbf{y}(k - 1) \dots \mathbf{y}(k - k_y)],$$

для которого вычисляется последующее значение выхода модели $y_m(k)$ как взвешенная сумма выводов правил:

$$y_m(k) = \sum_{i=1}^L \beta_i y_m^i(k), \quad (11)$$

где $0 \leq \beta_i \leq 1$ — весовой коэффициент i -го правила, определяющий степень соответствия реальных изменений вектора \mathbf{Z} изменениям, отраженным в i -ом правиле, и определяемый как

$$\beta_i(z) = \frac{\prod_{j=1}^Z A_{i,j}(z_j)}{\sum_{i=1}^L \prod_{j=1}^Z A_{i,j}(z_j)} \quad (12)$$

Таким образом, текущее значение выхода исправного объекта находится как скорректированное в соответствии с законом его функционирования предыдущее значение. Для неисправного объекта на анализируемой выборке переходного процесса будет происходить процесс накопления отклонений его выхода относительно выхода NARX-модели, построенной для исправного технического состояния объекта. Максимальное значение ошибки $\delta_{\text{макс}}$, определенное в соответствии с (1), зависит от точности настройки модели на выборках переходных режимов работы при исправном техническом состоянии объекта. Превышение среднеквадратичного отклонения выхода реального объекта в текущем переходном процессе $\delta_{\text{тек}}$ относительно максимально значения $\delta_{\text{макс}}$ должно формировать сообщение о возникновении неисправности объекта. Критерий определения неисправного технического состояния объекта может быть сформулирован в виде

$$\delta_{\text{тек}} \geq K_D \cdot \delta_{\text{макс}} = \delta_{\text{доп}}, \quad (13)$$

где K_D — коэффициент достоверности модели, $1 \leq K_D \leq 2$, $\delta_{\text{доп}}$ — допустимая ошибка моделирования.

База правил NARX-модели (10) показывает, что параметры посылок разделяют все множество возможных предыдущих значений векторов входа $\mathbf{u}(k - i)$, $i = k_d, \dots, k_u$ и выхода $\mathbf{y}(k - j)$, $j = 1, \dots, k_y$ на нечеткие области рассматриваемых режимов работы объекта. Параметры вывода описывают поведение выхода модели $y_m(k)$ объекта в этих режимах.

При оценке технического состояния электрооборудования необходимо оценить изменение не только выхода, но и промежуточных переменных состояния x_i , отражающих техническое состояние его внутренних узлов и элементов. Это позволяет определить конкретный неисправный элемент электрооборудования, ускорить ремонт и последующий ввод в работу. Объект может быть рассмотрен как последовательное соединение нечетких моделей, рис. 3, где оператор преобразования F_j , $j = 1, \dots, m$, пояснен на рис. 2.

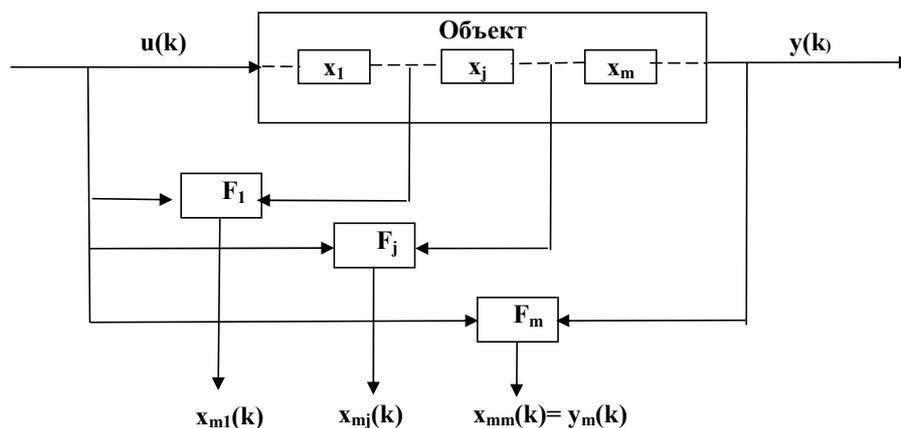


Рис. 3. Оценка технического состояния объекта на основе последовательности NARX-моделей.

Настройка NARX-модели заключается в определении параметров посылок, т. е. отклонений σ_u и σ_y гауссовых функций принадлежности, диапазонов изменения $(u_{\min} \div u_{\max})$, $(y_{\min} \div y_{\max})$ Z и S функций принадлежности и параметров вывода $(a_{i1} \dots a_{jku})$, $(b_{i1} \dots b_{jky})$, c_i , $i = 1, \dots, L$. Значения параметров входов и выходов в выборках переходных процессов используются для создания обучающих и проверочных данных при настройке адаптивной нейро-нечеткой сети ANFIS, реализующей NARX-модель для оценки технического состояния объекта. ANFIS представляет собой нейронную сеть прямого распространения, содержащую адаптивные узлы, выход которых зависит от их параметров. Правила обучения настраивают эти параметры так, чтобы минимизировать ошибку (1). Применяется гибридный обучающий алгоритм, каждая эпоха которого состоит из прямого и обратного оптимизационного расчетов. При прямом расчете исходная информация о значениях векторов входа $u(k-i)$, $i = k_d, \dots, k_u$ и выхода $y(k-j)$, $j = 1, \dots, k_y$ используется для определения методом наименьших квадратов (МНК) параметров вывода (a_{i1}, \dots, a_{jku}) , (b_{i1}, \dots, b_{jky}) , c_i , $i = 1 \dots L$, после чего рассчитывается ошибка ANFIS-сети. При обратном расчете методом градиентного спуска определяются параметры посылок, минимизирующие среднеквадратичное отклонение модели.

Ниже рассматриваются созданные NARX-модели АРВ и силового оборудования системы возбуждения.

NARX–модель АРВ для оценки его технического состояния, основана на структурной схеме рис. 4., где v –случайные воздействия, вызывающие переходные режимы работы синхронного генератора, w –возможные помехи, возникающих при измерении его параметров u и влияющие на выход АРВ $y(U_A)$.

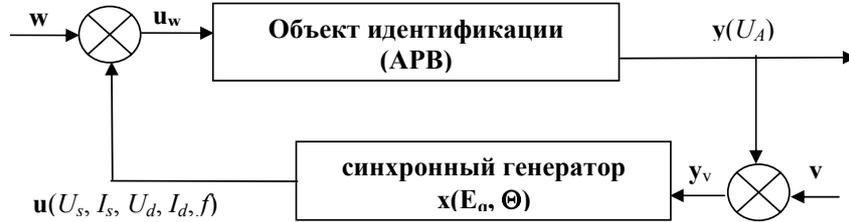


Рис. 4 Структурная схема АРВ для создания NARX–модели.

Анализ изменения выхода АРВ в переходном процессе для оценки его технического состояния требует последовательной обработки искаженного выхода АРВ y_v и вектора входов $u_w = u + w$. Выход АРВ для $(n+k)$ –ой точки зарегистрированной выборки переходного процесса $y_v(n+k)$ определяется как скорректированное в соответствии с законом регулирования значение текущего выхода АРВ в n -ой точке переходного процесса $y_v(n)$. Реализация последовательной обработки измерений выборки при оценивании выхода АРВ может быть интерпретирована как нелинейная авторегрессионная модель с внешним входом (NARX–модель). База нечетких правил модели АРВ представлена в табл. 2.

Таблица 2

U_{sk}	Rate_U	I_{sk}	Rate_f	U_{dk}	Rate_I _d	$U_A(k)$	$U_{Am}(k+n)$
not H	not H	not L	not L	not L	not L	not H	Z1
L	L	H	H	H	H	L	Z2
N	N	N	N	N	N	N	Z3
H	H	L	L	L	L	H	Z4
not L	not L	not H	not H	not H	not H	not L	Z5
H	N	L	N	VH	N	L	Z6

Примечание: RATE_U = $U_{s(k+1)} - U_{s(k)}$, RATE_f = $f_{(k+1)} - f_{(k)}$, RATE_I_d = $I_{d(k+1)} - I_{d(k)}$

Выводы каждого правила Z₁-Z₆ являются линейной комбинацией входных параметров (в соответствии с законом регулирования АРВ и его значением в предыдущий момент времени) и свободной составляющей и определяют соответственно действие АРВ при: Z₁ — форсировке системы возбуждения; Z₂ — установившемся режиме форсировки; Z₃ — начальном установившемся режиме; Z₄ — установившемся режиме расфорсировки; Z₅ — расфорсировке системы возбуждения; Z₆ — ограничении форсиров-

ки при превышении напряжением ротора двукратного номинального значения и определяются как:

$$\begin{aligned}
 Z_1 &= k_{11}U_s + k_{12}(RATE_U_s) + k_{13}I_s + k_{14}(RATE_f) + k_{15}U_d + k_{16}I_d + k_{17}U_A + m_1, \\
 Z_2 &= k_{21}U_s + k_{22}(RATE_U_s) + k_{23}I_s + k_{24}(RATE_f) + k_{25}U_d + k_{26}I_d + k_{27}U_A + m_2, \\
 Z_3 &= k_{31}U_s + k_{32}(RATE_U_s) + k_{33}I_s + k_{34}(RATE_f) + k_{35}U_d + k_{36}I_d + k_{37}U_A + m_3, \\
 Z_4 &= k_{41}U_s + k_{42}(RATE_U_s) + k_{43}I_s + k_{44}(RATE_f) + k_{45}U_d + k_{46}I_d + k_{47}U_A + m_4, \\
 Z_5 &= k_{51}U_s + k_{52}(RATE_U_s) + k_{53}I_s + k_{54}(RATE_f) + k_{55}U_d + k_{56}I_d + k_{57}U_A + m_5, \\
 Z_6 &= k_{61}U_s + k_{62}(RATE_U_s) + k_{63}I_s + k_{64}(RATE_f) + k_{65}U_d + k_{66}I_d + k_{67}U_A + m_6,
 \end{aligned}
 \tag{14}$$

Для создания последовательности NARX-моделей силового оборудования системы возбуждения использован граф причинно-следственных связей, рис. 5, на котором обозначены: U_A — выход АРВ, I_{1c} — переменный ток тиристорного преобразователя, u_f — напряжение тиристорного преобразователя, приложенное к обмотке возбуждения возбудителя, i_f — ток обмотки возбуждения возбудителя, I_E — ток якоря обращенного синхронного генератора, U_d — напряжение обмотки возбуждения, I_d — ток обмотки возбуждения, ВТ — выпрямительный трансформатор, ТП — тиристорный преобразователь, СУТ — система управления тиристорами, ОБВ — обмотка возбуждения возбудителя, ОСГ — обращенный синхронный генератор, ВВ — вращающийся выпрямитель, ЩКА — щеточно-контактный аппарат, ОБ — обмотка возбуждения турбогенератора.

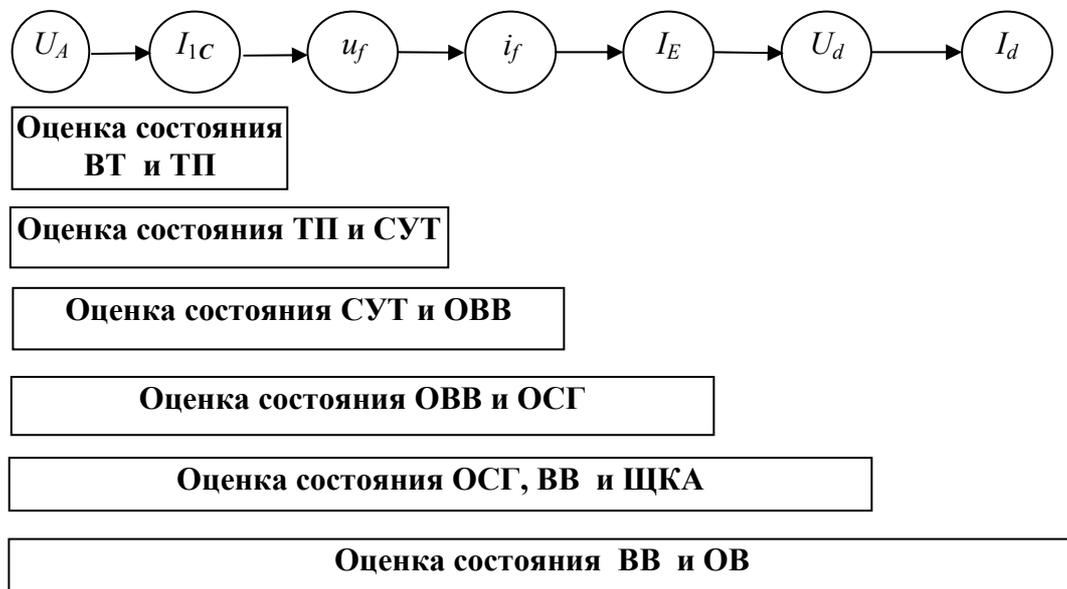


Рис. 5. Граф причинно-следственных связей оборудования системы возбуждения.

Создана следующая последовательность NARX-моделей:

1. $I_{1c}(k) = F\{I_{1c}(k-1), U_A(k-1)\}$ — оценка технического состояния цепей выпрямительного трансформатора и тиристорного преобразователя;

2. $u_f(k) = F\{u_f(k-1), I_{1c}(k-1), U_A(k-1)\}$ — оценка технического состояния цепей тиристорного преобразователя и системы управления тиристорами;

3. $i_f(k) = F\{i_f(k-1), u_f(k-1), I_{1c}(k-1), U_A(k-1)\}$ — оценка технического состояния системы управления тиристорами и цепей обмотки возбуждения возбудителя;

4. $I_E(k) = F\{I_E(k-1), i_f(k-1)\}$ — оценка технического состояния обмотки возбуждения возбудителя и обращенного синхронного генератора;

5. $U_d(k) = F\{U_d(k-1), I_E(k-1), i_f(k-1)\}$ — оценка технического состояния обращенного синхронного генератора, вращающегося выпрямителя и щеточно-контактного аппарата;

6. $I_d(k) = F\{I_d(k-1), U_d(k-1), I_E(k-1), i_f(k-1)\}$ — оценка технического состояния вращающегося выпрямителя и цепей обмотки возбуждения турбогенератора.

В табл. 3–8 представлены нечеткие правила NARX-моделей, отражающие базовые классы режимов работы системы возбуждения.

Таблица 3.

$I_{1c}(k-1)$	$U_A(k-1)$	$I_{1c}(k)$
not H	not L	$Z1 = b_{11}I_{1c}(k-1) + a_{11}U_A(k-1) + c_1$
N	N	$Z2 = b_{21}I_{1c}(k-1) + a_{21}U_A(k-1) + c_2$
not L	not H	$Z3 = b_{31}I_{1c}(k-1) + a_{31}U_A(k-1) + c_3$

Таблица 4.

$u_f(k-1)$	$I_{1c}(k-1)$	$U_A(k-1)$	$u_f(k)$
not H	not H	not L	$Z1 = b_{11}u_f(k-1) + a_{11}I_{1c}(k-1) + a_{12}U_A(k-1) + c_1$
N	N	N	$Z2 = b_{21}u_f(k-1) + a_{21}I_{1c}(k-1) + a_{22}U_A(k-1) + c_2$
not L	not L	not H	$Z3 = b_{31}u_f(k-1) + a_{31}I_{1c}(k-1) + a_{32}U_A(k-1) + c_3$

Таблица 5.

$i_f(k-1)$	$u_f(k-1)$	$I_{1c}(k-1)$	$U_A(k-1)$	$i_f(k)$
not H	not H	not H	not L	$Z1 = b_{11}i_f(k-1) + a_{11}u_f(k-1) + a_{12}I_{1c}(k-1) + a_{13}U_A(k-1) + c_1$
N	N	N	N	$Z2 = b_{21}i_f(k-1) + a_{21}u_f(k-1) + a_{22}I_{1c}(k-1) + a_{23}U_A(k-1) + c_2$
not L	not L	not L	not H	$Z3 = b_{31}i_f(k-1) + a_{31}u_f(k-1) + a_{32}I_{1c}(k-1) + a_{33}U_A(k-1) + c_3$

Таблица 6.

$I_E(k-1)$	$i_f(k-1)$	$I_E(k)$
not H	not H	$Z1 = b_{11}I_E(k-1) + a_{11}i_f(k-1) + c_1$
N	N	$Z2 = b_{21}I_E(k-1) + a_{21}i_f(k-1) + c_2$
not L	not L	$Z3 = b_{31}I_E(k-1) + a_{31}i_f(k-1) + c_3$

Таблица 7.

$U_d(k-1)$	$I_E(k-1)$	$i_f(k-1)$	$U_d(k)$
not H	not H	not H	$Z1=b_{11}U_d(k-1) + a_{11}I_E(k-1) + a_{12}i_f(k-1) + c_1$
N	N	N	$Z2=b_{21}U_d(k-1) + a_{21}I_E(k-1) + a_{22}i_f(k-1) + c_2$
not L	not L	not L	$Z3=b_{31}U_d(k-1) + a_{31}I_E(k-1) + a_{32}i_f(k-1) + c_3$

Таблица 8.

$I_d(k-1)$	$U_d(k-1)$	$I_E(k-1)$	$i_f(k-1)$	$I_d(k)$
not H	not H	not H	not H	$Z1=b_{11}I_d(k-1) + a_{11}u_f(k-1) + a_{12}I_{1c}(k-1) + a_{13}U_d(k-1) + c_1$
N	N	N	N	$Z2=b_{21}i_f(k-1) + a_{21}u_f(k-1) + a_{22}I_{1c}(k-1) + a_{23}U_d(k-1) + c_2$
not L	not L	not L	not H	$Z3=b_{31}i_f(k-1) + a_{31}u_f(k-1) + a_{32}I_{1c}(k-1) + a_{33}U_d(k-1) + c_3$

При недостаточной точности прогнозирования в некоторых эксплуатационных режимах каждая из моделей может быть дополнена правилами, отражающими параметры посылок в этих режимах.

NARX–модели оборудования системы возбуждения созданы с использованием пакета прикладных программ Fuzzy Logic Toolbox версии 2.1.2 вычислительной системы MATLAB 6.5 R13 [8]. Каждая модель состоит из двух файлов. Файл «Наименование модели. fis» представляет адаптивную нейро-нечеткую сеть модели. В файле заданы параметры посылок в виде символьных нечетких переменных, выходная переменная и база правил. Файл-сценарий «Наименование модели. m» на основании информации об изменении параметров системы возбуждения в выделенной выборке переходного процесса, представленной в формате COMTRADE, создает обучающие и проверочные данные для настройки сети. На основании сформированных данных гибридным методом обучения определяются оптимальные значения параметров предпосылок и коэффициентов выводов правил.

Действующие значения напряжения и тока статора синхронного генератора, используемые в правилах логического вывода, определяются по их мгновенным значениям, представленным в формате COMTRADE. Используется формула численного интегрирования Ньютона-Котеса [9]:

$$g^2(n) = \frac{U_a^2(n) + U_b^2(n) + U_c^2(n)}{3}; \quad (15)$$

$$\frac{1}{T} \int_{n-T}^n g^2(n) dn = \frac{1}{8} [g^2(n - \frac{3}{4}T) + 3g^2(n - \frac{1}{2}T) + 3g^2(n - \frac{1}{4}T) + g^2(n)],$$

где $T=20$ мс — период промышленной частоты.

Далее приведен пример настройки NARX–модели 5 (табл. 7). Сформулированы три нечетких правила, прогнозирующие значение напряжения ротора турбогенератора в зависимости от изменения тока возбуждения возбудителя, тока обмотки якоря обращенного синхронного генератора и предыдущего значения напряжения ротора при исправном техническом

состоянии обращенного синхронного генератора, вращающегося выпрямителя и щеточно-контактного аппарата. Каждое из правил отражает уменьшение, увеличение или неизменность текущего значения напряжения ротора относительно начального установившегося режима работы при изменении параметров обращенного синхронного генератора. При определении принадлежности параметров посылок в виде символьных нечетких переменных используется логический оператор дополнения. Первое правило формулируется в виде: «ЕСЛИ значение напряжения ротора $U_d(k-1)$ в предыдущей точке выборки переходного процесса НЕ принадлежит терму Н И значение тока обмотки якоря обращенного синхронного генератора $I_E(k-1)$ в предыдущей точке выборки переходного процесса НЕ принадлежит терму Н И значение тока возбуждения возбудителя обращенного синхронного генератора $i_f(k-1)$ в предыдущей точке выборки переходного процесса НЕ принадлежит терму Н ТО прогнозируемое значение напряжения ротора $U_d(k) = b_{11}U_d(k-1) + a_{11}I_E(k-1) + a_{12}i_f(k-1) + c_1$ ». В данном нечетком правиле сформулировано определение уменьшения напряжения ротора в переходном процессе относительно начального установившегося режима работы. Подобным образом, с учетом указанных в табл. 7 логических операторов, формулируются другие правила. При определении правил используется допущение о том, что прогнозируемая переменная не может измениться скачком относительно предыдущего значения, т. е. представляется в виде гладкой непрерывной функции. Это достигается логическими условиями, при которых предыдущее значение прогнозируемой величины в части посылок принадлежит формулируемому в правиле изменению. Если правило определяет уменьшение выходной переменной модели, то ее предыдущее значение не может принадлежать терму Н. Если правило определяет увеличение выходной переменной модели, то ее предыдущее значение не может принадлежать терму L.

Структура полученной ANFIS-сети в графическом редакторе пользователя пакета Fuzzy Logic Toolbox показана на рис. 6.

Рассмотрим назначение слоев сети.

Слой 1. Каждый узел является адаптивным с передаточной функцией $O_i^1 = \mu_{A_i}(x)$, где x — вход узла i , A_i — лингвистическая «нечеткая» переменная, ассоциированная с данным узлом. Для термина Н выбраны Гауссовы функции принадлежности

$$\mu_{A_i}(x) = \exp\left(-\frac{(x - l_i)^2}{2\sigma_i^2}\right), \quad (16)$$

S-образной формы для термина Н

$$\begin{aligned} \mu_{A_i}(x) &= 0, x \leq l_i; \\ \mu_{A_i}(x) &= 2 \left(\frac{x - l_i}{n_i - l_i} \right)^2, l_i < x \leq \frac{l_i + n_i}{2}; \\ \mu_{A_i}(x) &= 1 - 2 \left(\frac{n_i - x}{n_i - l_i} \right)^2, \frac{l_i + n_i}{2} < x \leq n_i; \end{aligned} \tag{17}$$

$$\mu_{A_i}(x) = 1, n_i < x$$

и Z-образной формы для термина L [5]

$$\mu_{A_i}(x) = 1, x \leq m_i;$$

$$\mu_{A_i}(x) = 1 - 2 \left(\frac{x - m_i}{l_i - m_i} \right)^2, d_i < x \leq \frac{l_i + m_i}{2}; \tag{18}$$

$$\mu_{A_i}(x) = 2 \left(\frac{l_i - x}{l_i - m_i} \right)^2, \frac{l_i + m_i}{2} < x \leq l_i;$$

$$\mu_{A_i}(x) = 0, l_i < x,$$

где $\{\sigma_i, m_i, l_i, n_i\}$ — множество настраиваемых параметров предпосылок.

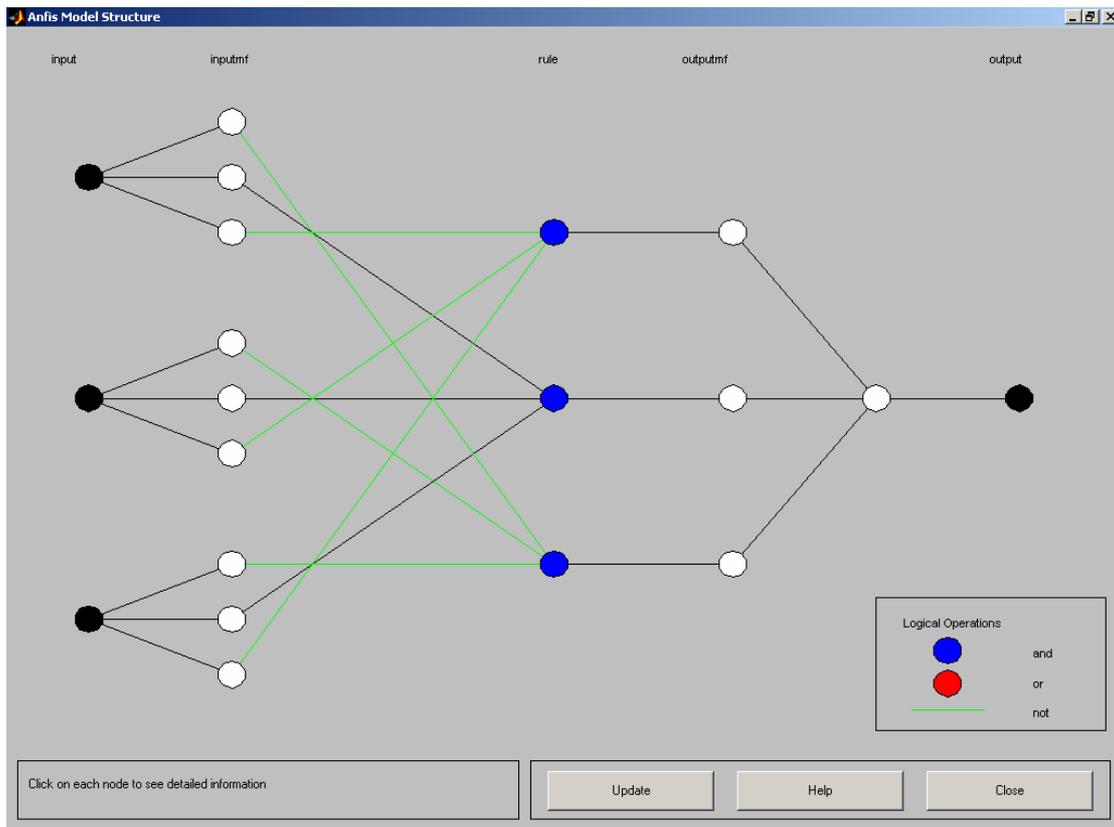


Рис. 6 Структура ANFIS-сети модели 5 (табл. 7).

Слой 2. Каждый узел данного слоя — неадаптивный, выполняющий логическую операцию И на параметрах посылок каждого правила. Выход этого узла — вес правила w_i определяется как произведение значений функций принадлежности $w_i = \mu_{U_d(k-1)i}(U_d) \times \mu_{I_E(k-1)i}(I_E) \times \mu_{I_f(k-1)i}(I_f)$, $i = 1, 2, 3$. Выход каждого узла этого слоя определяет степень срабатывания правила, т. е. выполнения условий, заложенных в параметрах предпосылок.

Слой 3. Неадаптивные узлы этого слоя вычисляют нормализованный вес правил:

$$\bar{w}_i = \frac{w_i}{w_1 + w_2 + w_3}, \quad i = 1, 2, 3. \quad (19)$$

Слой 4. Адаптивные узлы этого слоя имеют передаточные функции

$$O_i^4 = \bar{w}_i z_i = \bar{w}_i (b_{i1} U_d(k-1) + a_{i1} I_E(k-1) + a_{i2} I_f(k-1) + c_i), \quad i = 1, 2, 3, \quad (20)$$

где $\{a_{i1}, a_{i2}, b_{i1}, c_i\}$ — параметры вывода.

Слой 5. Неадаптивный узел этого слоя представляет собой сумматор

$$Z = O_1^5 = \sum \bar{w}_i z_i = \frac{\sum w_i z_i}{\sum w_i}. \quad (21)$$

Настройка коэффициентов NARX-модели осуществлялась на объединенной выборке 5 переходных процессов, возникших при приблизительно одинаковых начальных условиях. В выборку вошли переходные процессы, зарегистрированные в процессе эксплуатации:

- форсировка возбуждения при КЗ с успешным АПВ;
- форсировка возбуждения при КЗ с неуспешным АПВ;
- синхронные качания;
- процесс включения генератора в сеть;
- расфорсировка возбуждения.

Рассматриваемые выборки переходных процессов имели размер от 1200 до 10000 точек, что при частоте дискретизации АЦП микропроцессорного регистратора 2кГц соответствует длительности рассматриваемых переходных процессов 0.6 — 5с. Время обучения модели с использованием компьютера PIII-450 МГц с операционной системой Windows XP Professional и установленной вычислительной системой MATLAB 6.1.0.450 Release 12.1 составляет 2–3 минуты.

Исходя из технических характеристик систем возбуждения турбогенераторов ТВВ-1000 и проведенных вычислительных экспериментов с использованием гибридного обучающего алгоритма, определены параметры функций принадлежности посылок $\{\sigma_i, m_i, l_i, n_i\}$, табл. 9 и вывода $\{a_{i1}, a_{i2}, b_{i1}, c_i\}$:

$$\begin{aligned} Z1 &= 0.832 * U_d(k-1) - 9.159 * I_E(k-1) + 1.377 * I_f(k-1) - 129.6; \\ Z2 &= 0.75 * U_d(k-1) - 15.99 * I_E(k-1) + 0.452 * I_f(k-1) + 128.7; \\ Z3 &= 0.798 * U_d(k-1) - 4.06 * I_E(k-1) + 0.46 * I_f(k-1) - 36.86. \end{aligned} \quad (22)$$

Таблица 9.

Терм Параметр	L	N	H
$U_d(k-1), В$	$m_i = U_{d0} - 300, l_i = U_{d0}$	$\sigma_i = 100, l_i = U_{d0}$	$l_i = U_{d0}, n_i = U_{d0} + 300$
$I_E(k-1), кА$	$m_i = I_{E0} - 1, l_i = I_{E0}$	$\sigma_i = 2.6, l_i = I_{E0}$	$l_i = I_{E0}, n_i = I_{E0} + 1$
$i_f(k-1), А$	$m_i = i_{f0} - 100, l_i = i_{f0}$	$\sigma_i = 60, l_i = i_{f0}$	$l_i = i_{f0}, n_i = i_{f0} + 100$

Максимальная среднеквадратичная ошибка при обучении в соответствии с (1) составила 2.4 В. Для верификации модели на ее вход были поданы выборки 13 других зарегистрированных переходных процессов, не использующихся при ее обучении (КЗ, синхронные качания, расфорсировка возбуждения, форсировка возбуждения). Полученная максимальная среднеквадратичная ошибка в соответствии с (1) составила $\delta_{\max} = 2.5В$. Поэтому в соответствии с критерием (13) принято допустимое значение $\delta_{\text{доп}} = 3 В$. Если в текущем анализируемом переходном процессе $\delta_{\text{тек}} \geq \delta_{\text{доп}}$, то оборудование определяется как неисправное. Другие NARX-модели подобным образом были настроены в соответствии с критерием (13). Экспериментальные исследования на оборудовании бесщеточной диодной системы возбуждения турбогенераторов ТВВ-1000 Южноукраинской АЭС показали высокую эффективность предложенной методики: время оценки технического состояния в переходных режимах работы составляет 10–20 с при точности моделей, определяемых в соответствии с критерием (13).

Полученные модели исправного технического состояния оборудования являются одним из возможных решений поставленной задачи. Они могут быть положены в основу экспертной системы оценки технического состояния электрооборудования и поддержки принятия технических решений для использования в составе автоматизированной системы управления технологическими процессами энергоблока.

На рис. 7, 8 для сравнения, показаны изменения реального напряжения ротора и NARX-модели 5 в одном из зарегистрированных переходных процессов.

Файлы экспертной системы оценки технического состояния оборудования системы возбуждения и инструкции по их использованию приведены на сайте <http://www.MATLAB.ru/fuzzylogic/book1/index.asp> в разделе 4.

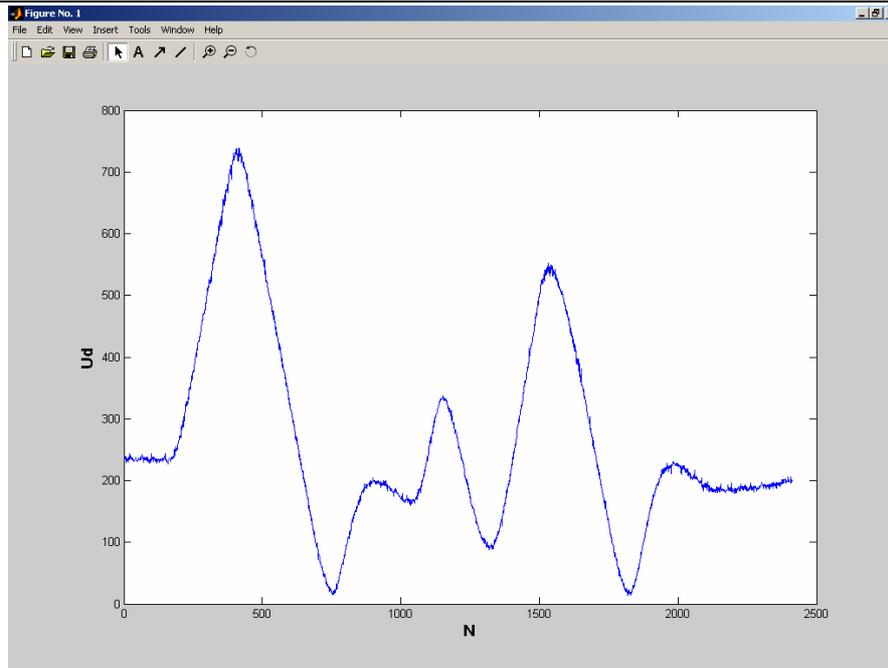


Рис. 7 График изменения напряжения ротора в одном из переходных процессов.

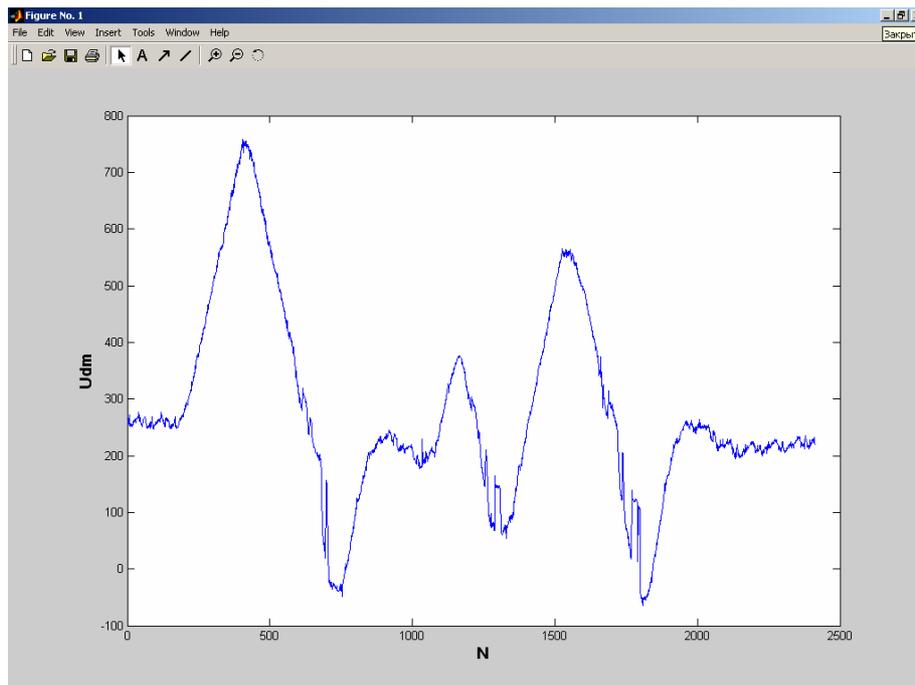


Рис. 8. График изменения выхода NARX–модели $U_d(k) = F\{U_d(k-1), I_B(k-1), i_f(k-1)\}$ в одном из переходных процессов.

Вывод

1. Предложена эффективная методика оценки технического состояния электрооборудования в реальном масштабе времени, основанная на последовательности NARX-моделей, реализуемых в виде адаптивных ней-

ро-нечетких сетей, обучаемых на выборках реальных переходных процессов исправного электрооборудования.

2. С использованием пакета MATLAB разработан прототип экспертной системы поддержки принятия решений при оценке технического состояния электрооборудования систем возбуждения мощных синхронных турбогенераторов типа ТВВ-1000 в реальном масштабе времени.

3. Внедрение методики повышает надежность работы оборудования электростанций, коэффициент использования установленной мощности и безопасность эксплуатации.

Литература

1. *Коллакот Р.* Диагностика повреждений. Пер. с англ.— М.: Мир, 1989.— 512 с.
2. *Алтунин А. Е., Семухин М. В.* Модели и алгоритмы принятия решений в нечетких условиях. Монография.— Тюмень: Изд-во Тюменского гос. университета, 2000.— 352 с.
3. *Агамалов О. Н., Костерев Н. В., Лукаш Н. П.* Оценка технического состояния систем возбуждения синхронных генераторов в реальном времени с использованием нечеткой логики // *Технічна електродинаміка. Тематичний випуск.*— 2002.— Ч.8.— С.24–29.
4. *Агамалов О. Н.* Оценка технического состояния электрооборудования в реальном времени методом нейро-нечеткой идентификации // *Электричество.*— 2003.— №7.— С.10–19.
5. *Дьяконов В, Круглов В.* Математические пакеты расширения MATLAB. Специальный справочник.— СПб: Питер, 2001.— 480 с.
6. *Jang J. S. R.* ANFIS: Adaptive network based fuzzy inference systems // *IEEE Trans. on Systems, Man, and Cybernetics.*— 23(03).— May 1993.— P.665–685.
7. *Abonji J., Babuska R., Verbruggen H. B., Szeifert F.* Incorporating Prior Knowledge in Fuzzy Model Identification // *International Journal of Systems Science.*— 31.— 2000.— P.657–667.
8. *Fuzzy Logic Toolbox. User's Guide, Version 2.*— The MathWorks, Inc., 1999.
9. Микропроцессорный комплекс для измерений параметров режима синхронной машины / *Маторный А.Ю., Нестерова Л.М., Трактовенко В. А.* // *Автоматизация исследований электрических машин и управление ими.*— Л.: Всесоюзный научно-исследовательский институт электромашиностроения, 1987.— С.90–99.

УДК 502.5:621.431

ПРИМЕНЕНИЕ MATLAB ПРИ РАЗРАБОТКЕ СИСТЕМ УПРАВЛЕНИЯ БЕНЗИНОВЫМИ ДВИГАТЕЛЯМИ ВНУТРЕННЕГО СГОРАНИЯ

*Аманов К. А., Воцанкин С. В., Смирнов А. Б., Черняк Б. Я.
Московский автомобильно-дорожный институт (ГТУ), Москва,
e-mail: alexy@server4you.ru*

На протяжении ряда лет на кафедре двигателей МАДИ (ГТУ) ведутся работы по совершенствованию алгоритмов микропроцессорных систем управления (МСУ) двигателей с искровым зажиганием (ДВС), и отработке технологии их калибровки.

Методически наши работы строились следующим образом. Разрабатывались модели процессов и работы контуров управления. На базе этих моделей велась отработка алгоритмов управления процессами или разработка технологии настройки системы управления. Найденные решения (при возможности) проверялись экспериментально на натуральных объектах.

Трудность решения последней задачи заключалась в том, что далеко не для всех решений удавалось реализовать натурные объекты и системы.

При настройке систем управления ДВС, оснащенных МСУ, чаще всего приходится сталкиваться с задачами моделирования процессов двигателя, идентификации моделей процессов, оптимизацией значений управляющих воздействий в системах двигателя и с аппроксимацией поверхностей параметров управления в зависимости от режима работы двигателя и выходных показателей. Наряду с этими задачами, исследователь сталкивается с проблемами синхронизации и фильтрации сигналов, полученных из различных измерительных систем испытательного стенда и из МСУ самого двигателя.

Исходя из специфики решаемых задач, можно сформулировать требования к интегрированной среде для моделирования процессов:

- сбор и обработка информации;
- возможность математических вычислений;
- разработка и исследование динамических моделей, как объекта управления, так и управляющих контуров;
- возможность исследования работы управляющей системы с динамическим функциональным аналогом объекта управления;
- наглядный вывод результатов моделирования;
- наличие оптимизационных процедур;
- открытость архитектуры;
- интеграция с офисными приложениями;

- возможность использования программных модулей, написанных на других языках программирования.

Всем указанным требованиям отвечает интегрированная среда математического моделирования MATLAB. Именно с этим связан наш переход от разработки программного обеспечения на языке Си++ к работе в среде MATLAB.

Рассмотрим несколько задач, которые были решены с использованием программных средств MATLAB.

1. Идентификация модели с помощью обычной математики

Для автомобильных двигателей требуется с очень высокой точностью поддерживать стехиометрический состав смеси ($\alpha=1.0$) в переходных процессах для обеспечения эффективной работы нейтрализатора. Поэтому одним из этапов калибровки современной МСУ является подбор параметров динамического корректора подачи топлива, основанного на моделях процессов образования смеси во впускном трубопроводе.

Пока в нашей стране подбор параметров корректора производится инженером-калибровщиком вручную, т. е. без использования автоматизированных поисковых процедур. Данный подход приемлем при подборе небольшого числа связанных параметров модели (2-4). При большом числе параметров (5 и более) такой подход приводит к значительным затратам времени. А подчас, человек уже не может справиться с решением задачи подбора оптимальных значений управляющих воздействий. В связи с этим была поставлена задача разработки автоматизированных процедур настройки корректора.

Первоначальным этапом было использование экспериментальных пошаговых оптимизационных процедур. Использовались эффективные оптимизационные процедуры типа МНСО, разработанные под руководством проф. д. т. н. Егорова И. Н. Решение находилось в результате ряда последовательных экспериментальных обращений к объекту управления с промежуточной расчетной оптимизацией. При числе параметров управления до 3, количество обращений к двигателю ограничивалось 15–20, что является приемлемым [1]. Но при 5–6 параметрах управления количество обращений к двигателю возрастало до 50–60, что является неудовлетворительным.

В связи с этим была поставлена задача исследования возможности настройки 6-ти параметрического динамического корректора подачи топлива по одной записи переходного процесса по составу смеси на основе идентификации модели переходного процесса. Эта задача решалась в рамках Межотраслевой программы сотрудничества Министерства образования Российской Федерации и АО «АВТОВАЗ». Для этого в SIMULINK была построена модель корректора, являющаяся копией реального коррек-

тора, заложенного в МСУ двигателя. На языке MATLAB была реализована оптимизационная процедура с использованием МНК для максимального приближения модели переходного процесса к реальной кривой. Управление моделью, процессом оптимизации и обработкой данных осуществлялось с помощью специального графического интерфейса.

Общая структурная схема программного оптимизационного комплекса представлена на рис. 1.

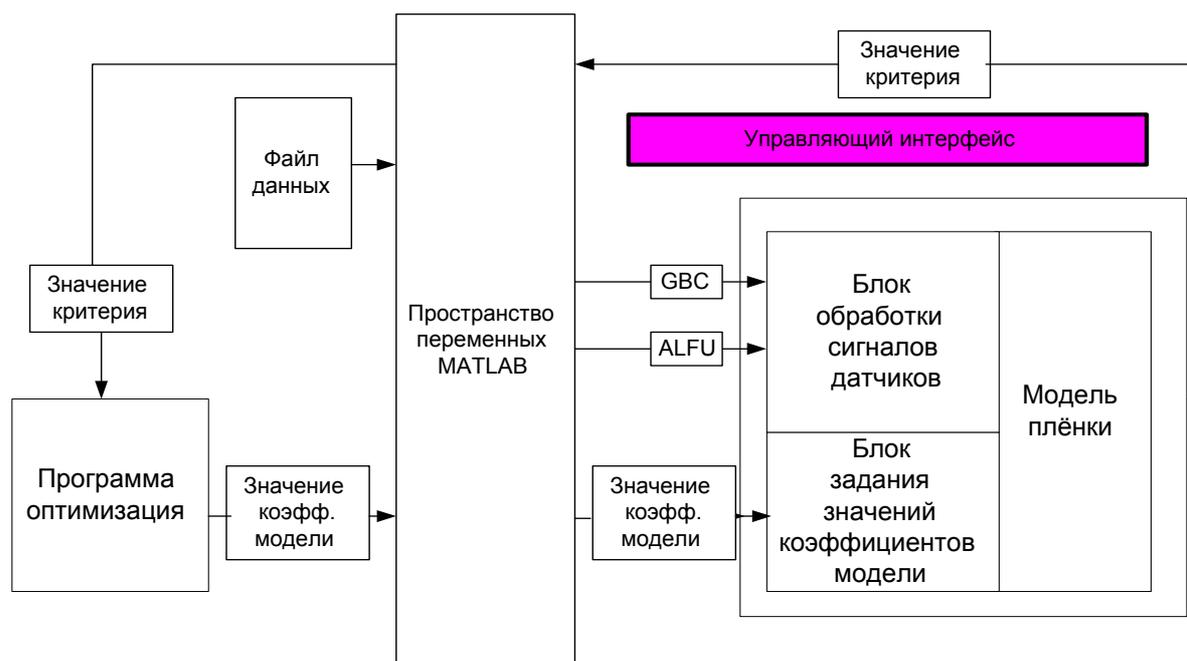


Рис. 1. Структура оптимизационного программного комплекса: GBC — цикловое наполнение двигателя, ALFU — реальный состав смеси в цилиндре.

На рис. 2 представлены исходные регистрации переходного процесса, полученные в лаборатории «НПП ЭЛКАР» и результаты его аппроксимации моделью. Основной задачей являлось описание переходного процесса после открытия дроссельной заслонки, который характеризуется резким обеднением смеси (увеличением α). Найденные решения (рис.2) в силу наличия значительных случайных возмущений имели ограниченную точность. Однако экспериментальная проверка на двигателе показала, что даже в этом случае удалось добиться примерно 3-х кратного уменьшения отклонения состава смеси в переходном процессе (табл.1). Для более точного решения можно использовать дополнительную итерацию.

Расчётная оптимизация



Рис.2. Исходные данные для идентификации модели и результаты описания переходного процесса с использованием оптимизационных процедур: желтая кривая — реальный состав смеси в цилиндре, фиолетовая кривая — модельный состав смеси.

Таблица 1.

Результаты проверки оптимизации параметров модели на двигателе.

Циклы увеличения нагрузки	Значения максимального обеднения состава смеси по каждому отдельному увеличению нагрузки			
	I	II	III	IV
α_{\max} до коррекции	1.09	1.09	1.09	1.07
α_{\max} после коррекции	1.03	1.03	1.02	1.03

2. Нейронный корректор

В качестве следующей задачи была принята разработка самообучающейся модели динамического корректора топливоподачи на основе искусственной нейронной сети (ИНС) [2]. Корректор имел четыре входа (цикловое наполнение, частота вращения и их изменение за цикл) и один выход, т. е. управлял непосредственно подачей добавочного топлива. Данная задача решалась с использованием только расчетного эксперимента на основе исходных регистраций процессов на двигателе.

Как видно из результатов, показанных на рис. 4, в исходном варианте при работе без корректора наблюдаются значительные отклонения состава смеси при открытии и закрытии дроссельной заслонки. После обучения нейронный корректор обеспечивал высокое постоянство состава смеси.

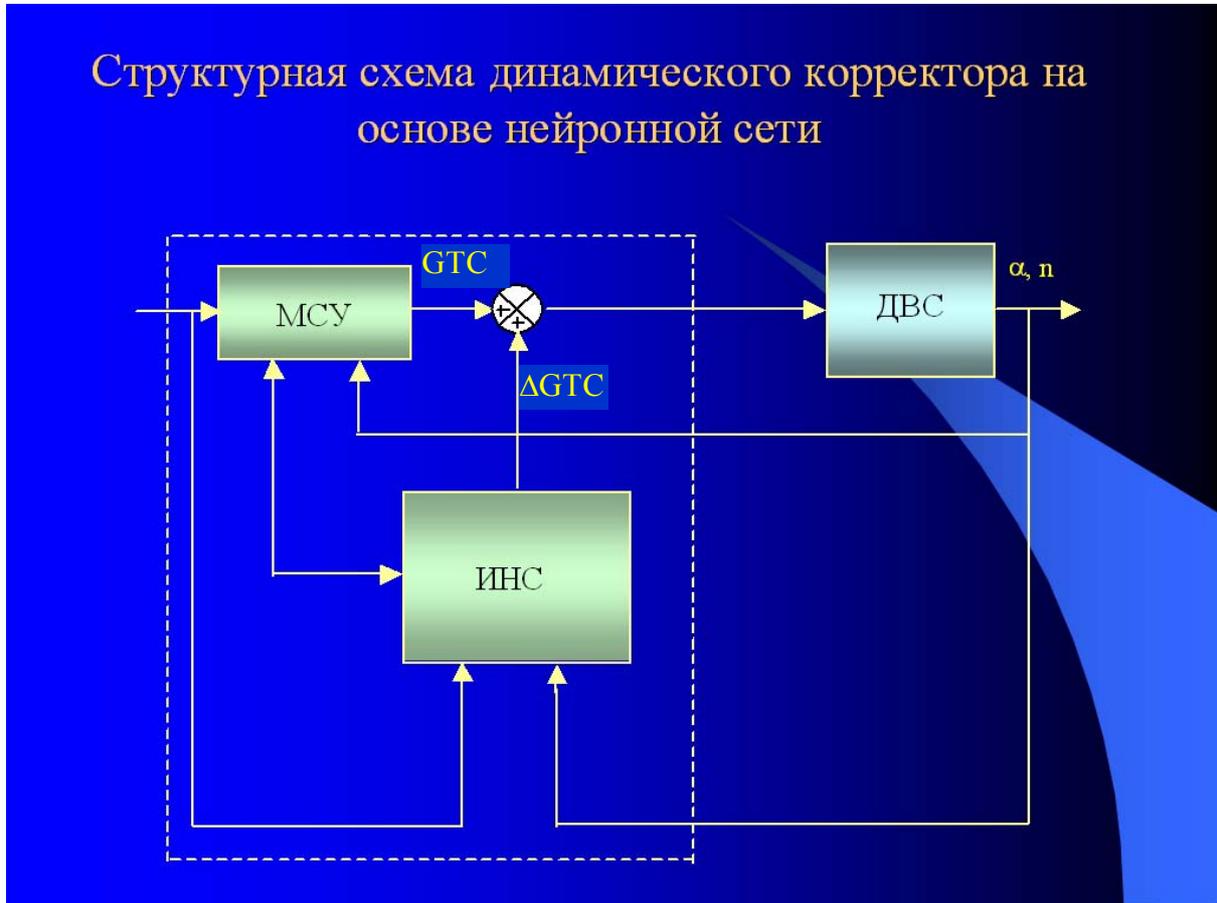


Рис. 3. Структурная схема динамического нейронного корректора топливоподачи.

Динамика основных показателей при резком изменении положения дроссельной заслонки без и с коррекцией

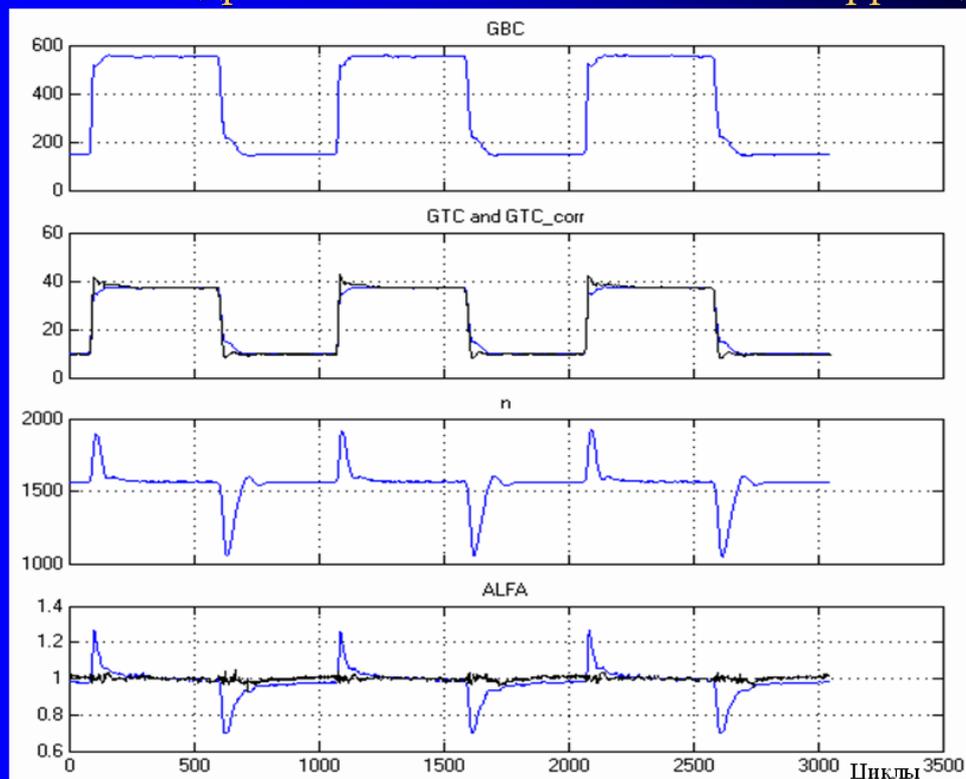


Рис.4. Результаты обучения нейронного корректора (где GBC — цикловое наполнение воздухом двигателя, GTC и GTC_corr — цикловые подачи топлива без и с коррекцией соответственно, n — частота вращения двигателя, ALFA — состав смеси в цилиндре до (синий) и после коррекции (черный)).

2. Аппроксимация параметра двигателя с помощью ИНС

Для того, чтобы отработать методику построения параметров корректоров в широком поле режимов по относительно малому числу опытов были опробованы процедуры аппроксимации результатов настроек на отдельных режимах. Пример решения такой задачи показан на рис. 5.

Опыты подтвердили возможность быстрой аппроксимации данных с достаточно высокой точностью. Среднеквадратичное отклонение на 21 точке не превышало 0.0286. Проверка погрешности прогнозирования не проводилась из-за отсутствия дополнительных экспериментальных данных.

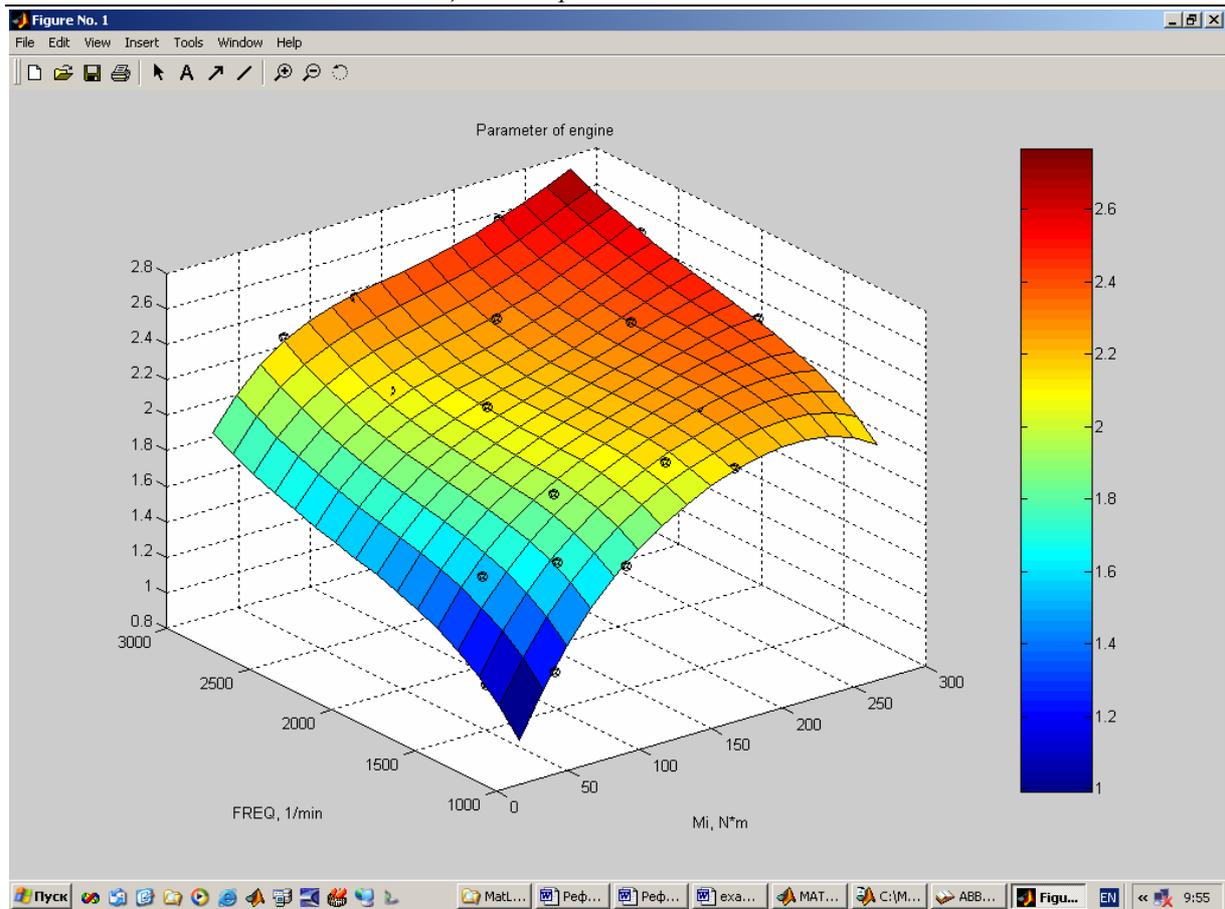


Рис. 5 Аппроксимация с помощью ИНС одного из параметров управления двигателем в поле нагрузочно — скоростных режимов.

Литература

1. Egorov I. N., Kretinin G. V., Pugayko A. N., Havtorin S. V., Chernyak B. Ja. Fast method of experimental calibrating of microprocessor control system // The European automotive industry meets the challenges of the year 2000. 5th International Congress.— Strasbourg, 21–23 June 1995.
2. Сигеру Омату, Марзуки Халид, Рубия Юсоф. Нейроуправление и его приложения. Нейрокомпьютеры и их применение (Книга 2): пер. с англ. Н. В. Батина, под ред. д. т. н. проф. А. И. Галушкина и д. т. н. проф. В. А. Птичкина.— М.: ИПРЖР, 2000.— 271 с.

УДК 519.7

ИСПОЛЬЗОВАНИЕ ПАКЕТА NEURAL NETWORK TOOLBOX СРЕДЫ MATLAB ДЛЯ ДИАГНОСТИКИ ГЕПАТИТА У ХИРУРГИЧЕСКИХ БОЛЬНЫХ

*Артюхин В. В., Соломаха А. А., Горбаченко В. И.
Пензенский государственный педагогический университет, Пенза,
Областная клиническая больница им. Н. Н. Бурденко, Пенза,
Пензенский государственный педагогический университет, Пенза,
e-mail:Scar@sura.ru, gorvi@mail.ru*

1. Введение

Актуальность ранней диагностики заболеваний, вызывающих анемию и повышающих риск неблагоприятных послеоперационных исходов, остается до сих пор нерешенной задачей [1]. Использование современных методов восполнения кровопотери: аутогемотрансфузии, аппаратной реинфузии крови, стимуляторов эритропоэза, имеет ограничение в клинической хирургии. Так, широкое применение аутогемотрансфузии невозможно из-за того, что имеют место наблюдения, когда в клинической практике у больных предоперационная заготовка аутокрови усугубляет течение анемии. Аппаратная реинфузия крови не может применяться в случаях бактериального инфицирования крови, излившейся в полости организма, когда повреждены полые органы, а также при наличии нагноительных заболеваний легких, плевры, брюшины. Стимуляторы эритропоэза эффективны при низком эндогенном уровне эритропоэтина в крови больных. Однако клинический эффект длителен и составляет 2–3 недели, в то же время тяжелую анемию они способны компенсировать неадекватно перед выполнением оперативного вмешательства. Стимуляторы эритропоэза успешно применяются у больных, находящихся на программном гемодиализе с хронической почечной недостаточностью [2]. Известно, что вызывать анемию могут сопутствующие основному заболеванию, например вирусные гепатиты.

Для проведения адекватной компенсации анемии важна ранняя диагностика сопутствующей патологии, повышающей риск ее возникновения [3].

Известные методы лабораторной диагностики: иммуноферментный анализ (ИФА) и полимеразно-цепная реакция (ПЦР) являются достоверными в диагностике вирусного гепатита [4]. Однако они редко применяются у больных с анемией без клинического проявления и жалоб больного на заболевание печени. ИФА и ПЦР являются дорогостоящими методами. Поэтому они не могут быть первыми для проведения скрининга.

Перспективным направлением является формализация задач медицинской диагностики, в частности, применение статистических и нейросетевых методов. Применение методов статистического анализа в рассматриваемой области ограничивается нечеткой трактовкой моделей нормы, адаптации и патологии [5]. Наиболее универсальным инструментом диагностики являются нейронные сети [6].

Целью работы является исследование нейронных сетей, позволяющих эффективно распознавать лабораторную характеристику вирусного гепатита с целью его скрининга, без привлечения дорогостоящих методов ИФА и ПЦР.

2. Выбор и подготовка исходных данных

С целью решения поставленной задачи сформированы две группы наблюдений: обучающая и контрольная. В качестве исходных данных использовались 7 параметров, которые характеризуют наличие гепатита у пациента: скорость оседания эритроцитов, общий белок, билирубин, аспартатамино-трансфераза, аланинамино-трансфераза, щелочная фосфатаза, тимоловая проба. Первую группу наблюдений составили 150 здоровых доноров отделения переливания крови, вторую — 150 больных вирусными гепатитами *B* и *C* в фазе репликации отделения гепатологии [7]. Таким образом, обучающая и контрольная выборки были составлены из 150 клинических наблюдений: по 75 доноров и пациентов отделения гепатологии.

Исходные данные представлялись и предварительно анализировались в электронных таблицах Excel. Обучающее множество подвергалось статистической обработке, исключались выбросы, отсутствующие анализы заменялись средними значениями. Нейронные сети моделировались в системе MATLAB, в которую таблицы исходных данных импортировались с помощью функции **xlsread**.

Результаты анализов представляют собой числа, резко различающиеся для разных видов анализов. Исходные данные приводились к интервалу $[-1\ 1]$. Для этого обучающее множество масштабировалось с использованием функции **premnmx**, все другие входные данные сетей масштабировались с помощью функции **trnmmx** [8, 9].

3. Выбор архитектуры сети

При построении нейронной сети важно выбрать тип сети, число слоев и число нейронов в каждом слое, размер обучающей выборки. До сих пор не существует аналитических методов выбора параметров нейронных сетей. Выбор архитектуры сети производится на основе опыта и экспериментов. Известны [10, 11] рекомендации по выбору архитектуры сети для аппроксимации функций. Для задач классификации, к которым относится

рассматриваемая задача, такие рекомендации отсутствуют. Эксперименты с сетями различной архитектуры показали, что для решения рассматриваемой задачи достаточно персептрона.

В экспериментах использовался персептрон с 7 входами, с симметричной ступенчатой функцией активации **hardlims** и функцией адаптации **adapt** [8, 9].

Так как выход сети изменяется в диапазоне $[-1\ 1]$, то необходимо принять правила интерпретации результатов.

4. Результаты экспериментов

Эксперименты показали, что для обучения персептрона потребовалось 5 проходов (**passes**). При этом на контрольной выборке обученный персептрон дает погрешность 4,67 %.

Заключение

Диагностика вирусного гепатита может успешно проводиться с помощью созданной нейронной сети. Применение этой технологии целесообразно для улучшения исходов оперативного лечения. Предлагаемая методика рекомендуется при проведении скрининга гепатита в хирургии.

Литература

1. Соломаха А. А. Исследования лабораторных показателей крови доноров // Вестник службы крови России.— 2003.— № 3.— С.23–25.
2. Соломаха А. А., Хрусталева Е. В. Компенсация анемии эритропоэтином в клинической трансфузиологии и нефрологии // Бескровная хирургия / Под ред. Ю. В. Таричко.— М.: Центр образовательной литературы. 2003.— С.40–41.
3. Соломаха А. А. Опыт работы трансфузиологической службы многопрофильной хирургической клиники г. Пенза // Новое в трансфузиологии. Вып. 35.— М., 2003.— С.74–77.
4. Соломаха А. А., Ледванов М. Ю., Егорова Е. В. Организация генотипирования вирусного гепатита С в службе крови многопрофильной больницы // Клиническая лабораторная диагностика.— 2003.— №9.— С.34.
5. Славин М. Б. Методы системного анализа в медицинских исследованиях.— М.: Медицина, 1989.— 304 с.
6. Горбань А. Н., Россиев Д. А. Нейронные сети на персональном компьютере.— Новосибирск: Наука, 1996.— 276 с.
7. Соломаха А. А., Емелина Л. А., Артюхин В. В. Метод дифференциальной диагностики вирусного гепатита в учреждениях службы крови и эндоэкологическое прогнозирование // III Всероссийская науч.-практ. конф. «Экология и ресурсоэнергосберегающие технологии на предприятиях

- народного хозяйства»: Сб. материалов.— 16–17 октября 2003.— С.164–166.
8. *Медведев В. С., Потемкин В. Г.* Нейронные сети. MATLAB 6.— М.: ДИАЛОГ-МИФИ, 2002.— 496 с.
 9. *Demuth H.* Neural Network Toolbox. User's Guide. Version 4 / H. Demuth, M. Beale.— MathWorks, Inc., 2001.— 844 p.
 10. *Осовский С.* Нейронные сети для обработки информации.— М.: Финансы и статистика, 2002.— 344 с.
 11. *Haykin S.* Neural Networks: a Comprehensive Foundation.— New Jersey: Prentice Hall, 1999.— 842 с.

УДК 519.711.3

РЕШЕНИЕ РЕСУРСОЕМКИХ ВЫЧИСЛИТЕЛЬНЫХ ЗАДАЧ С ПОМОЩЬЮ НЕЙРОСЕТЕЙ

Злобин В. В.

*Московский государственный университет им. М.В.Ломоносова,
факультет вычислительной математики и кибернетики, Москва,
e-mail:zlobin@cs.msu.su*

Математическое моделирование призвано заменить натурные эксперименты. Математическая модель стала единственным средством изучения явлений в тех областях человеческой деятельности, где проведение натуральных экспериментов связано с большими финансовыми затратами или риском для жизни людей. К такой области деятельности относится разработка установок для осуществления управляемого термоядерного синтеза (УТС) [1]. Целью моделирования УТС является увеличение длительности разряда в плазме. Увеличение достигается за счет выбора оптимальных параметров, поиск которых осуществляется путем оптимизации соответствующей математической модели. Но оказывается, что сами математические модели настолько сложны и требуют столько вычислительных ресурсов, что их невозможно использовать для поиска оптимальных параметров. Например, для кода SCoPE, моделирующего эволюцию равновесия тороидальной плазмы, время одного расчета занимает несколько часов на компьютере P4 1700MHz. Но число параметров, влияющих на длительность разряда — несколько десятков. Поэтому, предлагается заменить саму математическую модель еще более простой моделью на основе нейросетей. А как известно, одно прямое вычисление по нейросети выполняется за доли секунды. Эту нейросеть можно использовать как модель процессов, происходящих в плазме. Таким образом, авторами предлагается следующая методика.

1. Проводятся расчеты по исходной математической модели. Наборы соответствующих параметров выбираются случайно.
2. С помощью проведенных расчетов настраивается нейросеть прямого распространения.
3. Проводится поиск оптимальных значений параметров с помощью градиентных методов, используя нейросеть как целевую функцию.
4. В этих точках проводятся расчеты по исходной модели.
5. Результаты новых расчетов используются для уточнения настройки нейросети. Окончательные результаты оптимизации проверяются по точной модели.

Для программной реализации методики была выбрана система MATLAB 6.5, так как она предоставляет удобные средства для работы с

нейросетями (Neural Network Toolbox) и градиентными методами оптимизации (Optimization Toolbox).

Решение задачи оптимизации в токамаке

Рассмотрим в качестве примера задачу об увеличении дополнительного (бутстрап) тока в установке токамак и продемонстрируем на ней все преимущества предлагаемого подхода. Способ получения оптимальных параметров поясняется на рис.1.

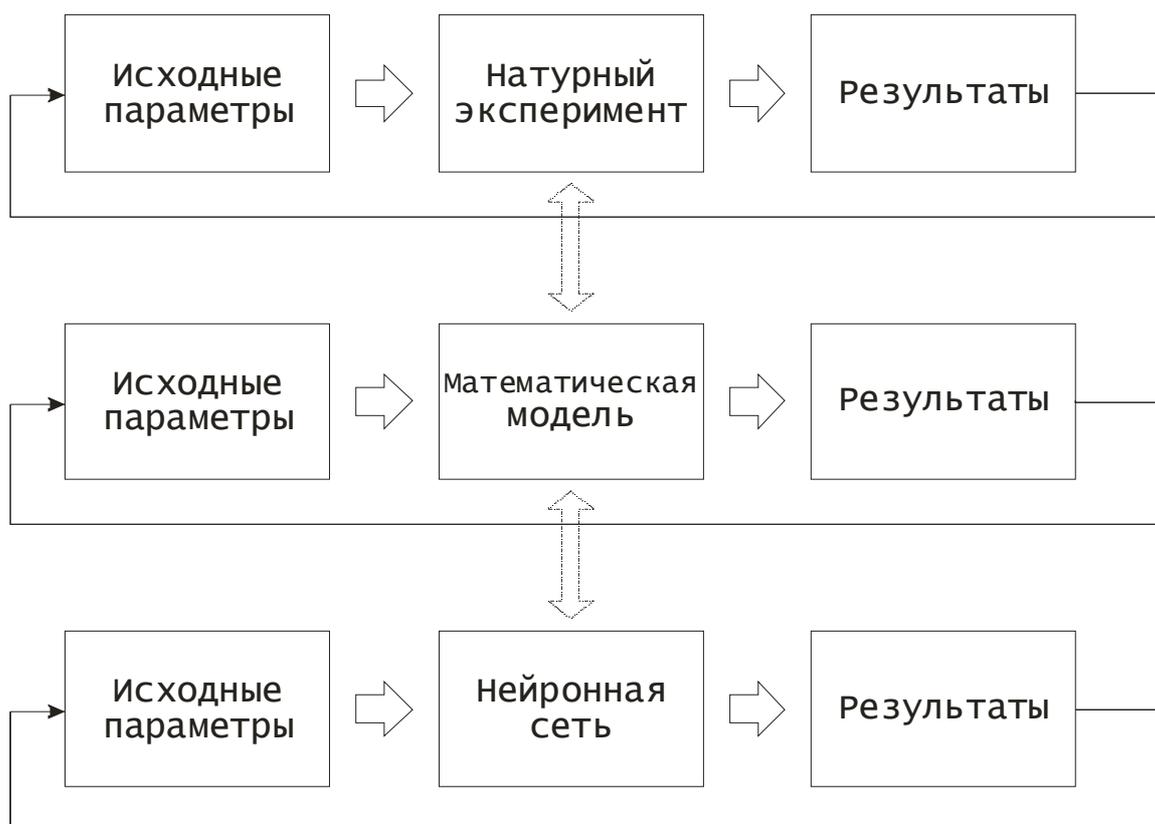


Рис. 1. Три сценария настройки параметров. Каждый переход на уровень ниже, расширяет возможности экспериментатора.

В первой строке приводится стандартная последовательность действий, которая применяется для настройки параметров некоторого процесса, например, физиком-экспериментатором. Задавая различные исходные данные, исследователь проводит полномасштабные натурные эксперименты. По результатам делаются выводы о степени влияния различных параметров, на основе чего выбираются их новые значения и эксперименты повторяются.

Вторая строка (рис.1) отражает переход от натурального эксперимента к его математической модели [2]. Здесь требуется проделать дополнительную работу по созданию адекватного математического описания иссле-

дуемого процесса. Затем, вместо исследования влияния параметров на процесс, проводится изучение математической модели, а зависимость от параметров исследуется на ее решении. Наряду с теоретическими методами изучения моделей, используется вычислительный эксперимент, суть которого состоит в численном решении возникающих при моделировании математических задач. В этом случае, исследователь совершает те же действия, что и физик-экспериментатор, за исключением того, что вместо проведения натурального эксперимента он исполняет программу на ЭВМ.

Нижняя строка на рис. 1 обозначает переход от математической модели, основанной на численных методах, к нейронной сети. На этом этапе исследователь создает нейронную сеть и обучает ее по результатам расчетов вычислительного эксперимента. Проверяет степень соответствия нейросети и исходной математической модели с помощью тестового множества параметров, не участвовавшего в обучении. И теперь, оптимальные значения параметров можно искать практически любым известным методом оптимизации, в том числе градиентными методами. Высокая скорость прямого вычисления значений нейросети (тысячи значений в секунду на современном персональном компьютере), позволяет воспользоваться даже перебором. В этом и заключается основное преимущество предлагаемой методики, потому что сложные математические модели не позволяют использовать эти методы поиска параметров из-за своей ресурсоемкости.

Процедуру увеличения дополнительного тока в установке токамак можно разбить на пять шагов, поясняющих рис. 2.

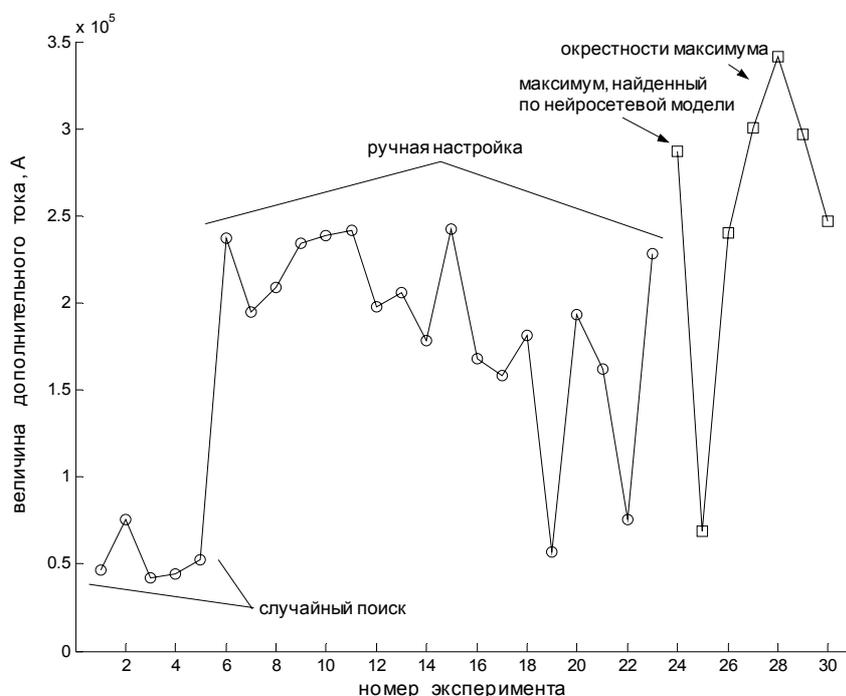


Рис. 2. Динамика изменения дополнительного тока в процессе оптимизации параметров. По оси X отложены порядковые номера экспериментов, а по оси Y — величина дополнительного тока в амперах.

Шаг 1: провести расчеты в случайных точках. Из постановки задачи было известно, что ток 50 кА (килоампер) считается хорошим значением. Среди первых пяти случайных точек, оказалась одна, с током 70 кА (см. рис.2). После этого, вместо случайных, выбирались такие точки, параметры которых были субъективно «похожи» на эту «эталонную» точку. Тут использовалась некоторая дополнительная информация о задаче, а именно, теоретическое положение о том, что большим значениям двух определенных параметров, соответствуют большие значения дополнительного тока. Но прямое применение этого правила невозможно, так как слишком большие значения параметров, несомненно, приведут к срыву плазмы, а значит, цель не будет достигнута. Тем не менее, среди следующих 17 точек, выбранных таким интуитивным образом, ток колебался около 200 кА, и максимум достиг 240 кА.

Дальнейшие попытки улучшить результат вручную, даже с использованием новых расчетных данных, не оправдались (рис.2, эксперименты с 6-го по 23-ий). Поэтому, переходим к следующему шагу.

Шаг 2: обучить нейросеть. Архитектура нейросети определяется заранее, но при увеличении размера обучающей выборки (по мере накопления результатов расчетов) может измениться. Например, может понадобиться увеличить число нейронов в скрытом слое, чтобы повысить точность нейросетевой модели. В качестве базовой использовалась следующая архитектура: два последовательно идущих скрытых слоя заканчиваются одним линейным выходным нейроном. Количество нейронов в каждом из скрытых слоев, выбиралось исходя из размеров обучающей выборки и ошибки на тестовом множестве. В конечном счете, наиболее удачной оказалась архитектура с четырьмя нейронами (по два в каждом скрытом слое) с функцией активации — гиперболический тангенс. Создание и обучение сети с такой архитектурой в MATLAB осуществляется двумя командами:

```
nn = newff([min(P') max(P')], [2 2 1], {'tansig', 'tansig', 'purelin'});  
nn = train(nn, P, T);
```

Где nn — переменная-нейросеть, P — входные параметры (каждый набор занимает столбец), T — строка со значениями целевой функции.

Для обучения нейросети использовался метод Левенберга-Маркварта. Поскольку обучающая выборка содержит меньше 50 точек, а нейросеть состоит лишь из 4 нейронов, то большие затраты памяти, приписываемые к недостаткам этого метода, практически не проявляются. Чтобы обучать сеть этим методом в MATLAB, перед обучением нужно добавить строку:

```
nn.trainFcn='trainlm';
```

Для остановки обучения использовалась байесовская регуляризация (*nn.trainFcn = 'trainbr'*) или проверка на неуменьшение тестовой ошибки в

течение 30 итераций (использовался параметр нейросети `nn.trainParam.max_fail`).

Был проведен ручной отбор обучающей и тестовой выборки из всех имеющихся данных. Обучив нейросеть, мы переходим к поиску максимума ее значений в той области, в которой мы ее обучали.

Шаг 3: Найти оптимальные значения параметров. Для оптимизации параметров по нейросетевой модели использовался квазиньютоновский алгоритм поиска экстремума (функция `fminunc`). Чтобы оценить реальную картину расположения экстремумов использовался метод мультистарта — поиск совершался из K начальных точек, при этом, максимальное число итераций квазиньютоновского алгоритма ограничивалось константой N . На практике использовались значения для K и N — около 30 и 50 соответственно.

Шаг 4: Провести расчеты в найденных точках. После отбора наилучших точек по нейросети, в этих точках были проведены дополнительные расчеты по исходной модели. В нашем случае был выбран только один максимум.

Как видно на рисунке (см. рис. 2, эксперимент 24, ток 280 кА), максимальное значение тока увеличилось на (18%). Это значимый результат, потому что к этому моменту, попытки улучшить ток вручную стали безуспешными. Помимо расчетов в этой точке, были также проведены расчеты в случайных точках из малой окрестности выбранного максимума. Ясно, что среди них могут оказаться еще лучшие результаты (как и оказалось в данной задаче, см. рис. 2, максимальный ток достиг 340 кА) — их можно будет использовать для следующих итераций по предложенной методике.

Шаг 5: Принять решение об остановке или повторить процедуру. На этом решение рассматриваемой задачи было прекращено. Конечный результат — увеличение дополнительного тока почти в 7 раз. К сожалению, проверка результатов по точной модели на данный момент не представляется возможной, хотя рассматривается такая возможность в перспективе.

Визуальная среда, реализованная в MATLAB

Для многократного применения данной методики была создана программа с графической оболочкой на основе пакета численных вычислений MATLAB 6.5. Она поддерживает все этапы, связанные с построением нейросетевой модели, выбором экстремальных значений и визуализацией данных. Кроме того, добавлена возможность визуализации многомерных данных с помощью сетей Кохонена. Для этого использовался дополнительный пакет для MATLAB (Som Toolbox), который распространяется бесплатно [3].

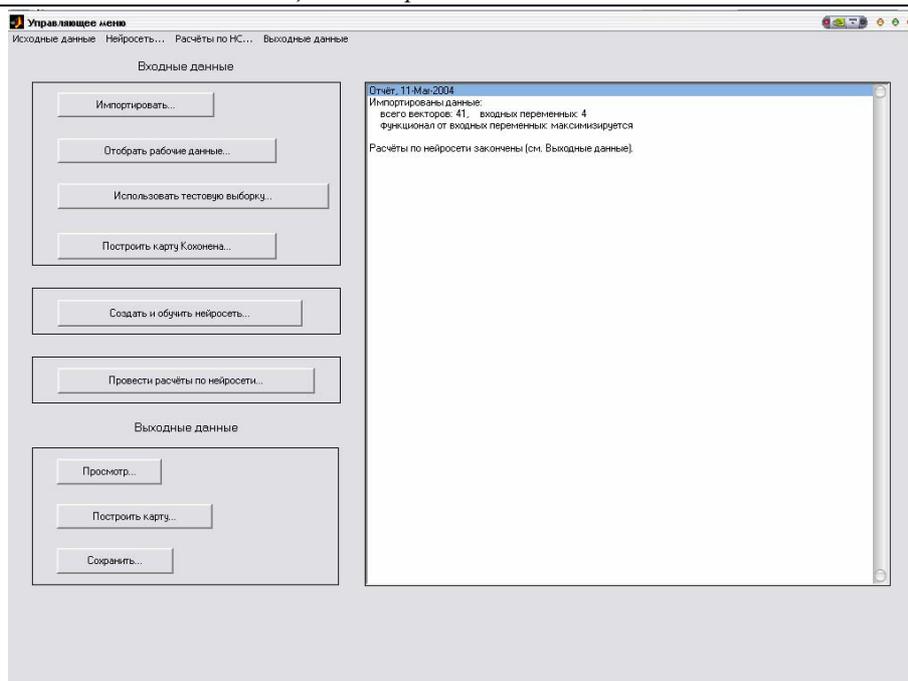


Рис. 4. Визуальная среда. Выделено четыре набора операций: действия с входными данными, с нейросетью, оптимизация и обработка выходных данных.

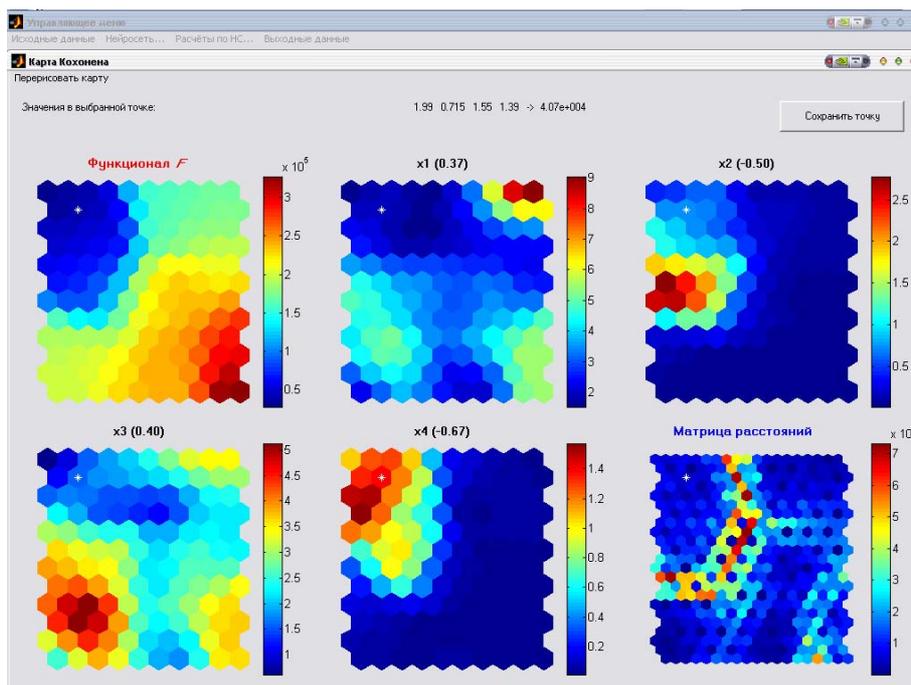


Рис. 3. Визуальная среда. Отображение целевого функционала, четырех входных параметров и матрицы расстояний по двумерной карте Кохонена, обученной на входных данных.

На сегодняшний день, нейросети успешно применяются для моделирования зависимостей, не имеющих аналитического выражения [4, 5]. В данной статье, приводятся пример, показывающий пользу применения нейросетей при работе с уже известными математическими моделями.

Также излагается общая схема применения нейронных сетей для оптимизации параметров со ссылками на функции, реализованные в системе MATLAB.

Предложенная методика полностью оправдала себя как с точки зрения существенной экономии машинных ресурсов и времени, затраченного на эксперименты, так и с точки зрения полученных результатов — удалось повысить величину дополнительного тока в 6.8 раза по сравнению с исходной величиной и на 29% по сравнению с максимумом, найденным вручную.

Автор выражает свою признательность проф. Ф. С. Зайцеву и А. А. Лукьянице за обсуждение настоящей работы.

Литература

1. *Днестровский Ю. Н., Костомаров Д. П.* Математическое моделирование плазмы.— М.: Наука, Гл.ред.физ.-мат.лит., 1982.— 320 с.
2. *Самарский А. А., Михайлов А. П.* Математическое моделирование: Идеи. Методы. Примеры.— М.: Физматлит, 2001.— 320 с.
3. <http://www.cis.hut.fi/projects/somtoolbox/>
4. *Magnus Nørsgaard, Charles C. Jorgensen, James C. Ross,* Neural Network Prediction of New Aircraft Design Coefficients // NASA Technical Memorandum 112197.— May 1997.
5. *Jürgen Van Gorp,* Steel Plant Modelling and Analysis Using a Neural Network // ANNIE '99, Intelligent Engineering Systems through Artificial Neural Networks.— November 1999.

УДК 519.7

СИСТЕМА ДИАГНОСТИКИ ШТАНГОВОЙ ГЛУБИННО-НАСОСНОЙ УСТАНОВКИ НА ОСНОВЕ НЕЙРОННОЙ СЕТИ

Зюзев А. М., Костылев А. В.

*Уральский государственный технический университет УПИ, Екатеринбург,
e-mail: zuzev@ep.etf.ustu.ru*

1. Постановка задачи

Штанговая глубинно-насосная установка является промышленным объектом, характер работы которого можно описать с помощью динамограммы, характеризующей зависимость усилия в подвеске полированного устьевого штока от перемещения или хода штока. При обработке динамограммы появляется возможность определить количественные и качественные показатели работы ШГНУ:

- нагрузки и напряжения в полированном штоке;
- длину хода плунжера и полированного штока;
- коэффициент наполнения насоса;
- герметичность приемной и нагнетательной частей насоса;
- влияние газа;
- правильность посадки плунжера;
- наличие утечек в нагнетательной колонне труб;
- отвороты и обрывы штанг или штанговых муфт;
- заклинивание плунжера.

По динамограмме работы станка-качалки в среде, содержащей свободный газ, также определяют давление у приема насоса, дебит жидкости и дебит газа.

Как правило, динамометрирование должны проводить в первый же день после спуска насоса в скважину и при изменениях режима откачки и подачи насоса, а также в процессе его работы для своевременного выявления различных неполадок. Для установления в каждом конкретном случае характера осложнений целесообразно воспользоваться типовыми динамограммами [1].

На рис. 1 представлены идеализированные динамограммы для основных состояний скважинного насоса, кроме аварийных. Исходя из приведенного набора типовых динамограмм, характерных для ШГНУ, при разработке системы диагностирования ставится задача распознавания указанных динамограмм средствами нейронной сети.

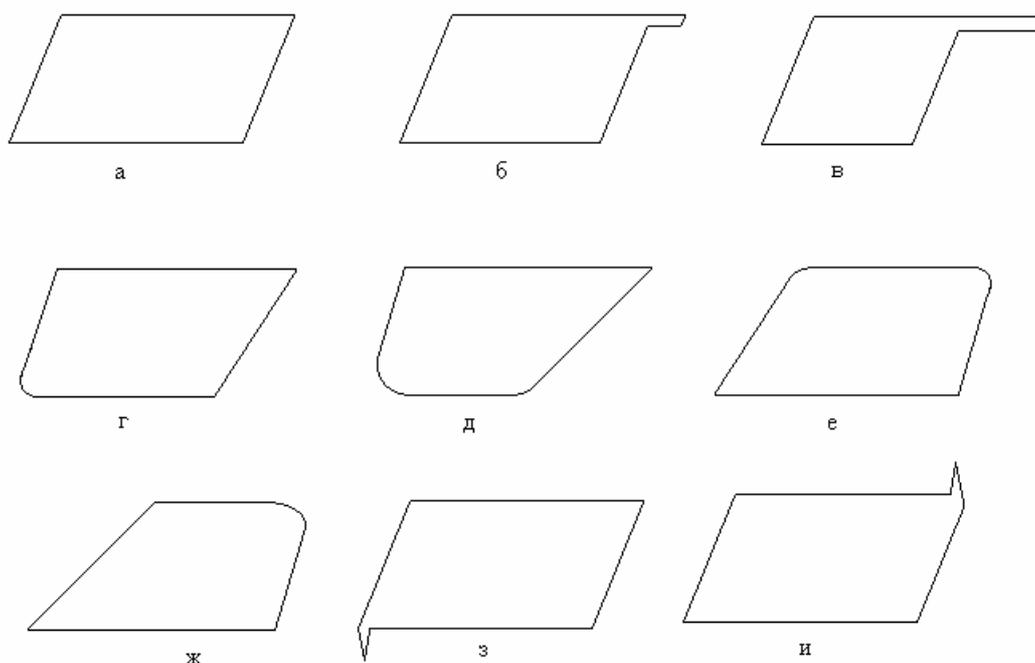
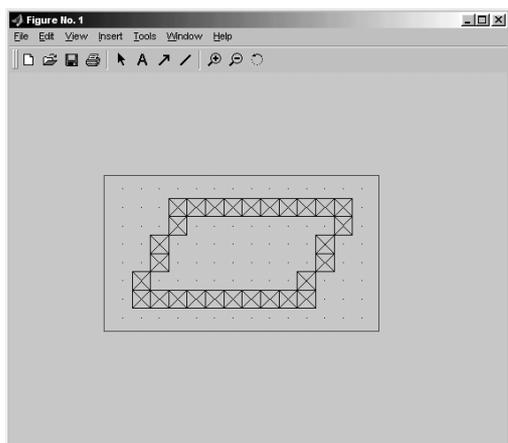


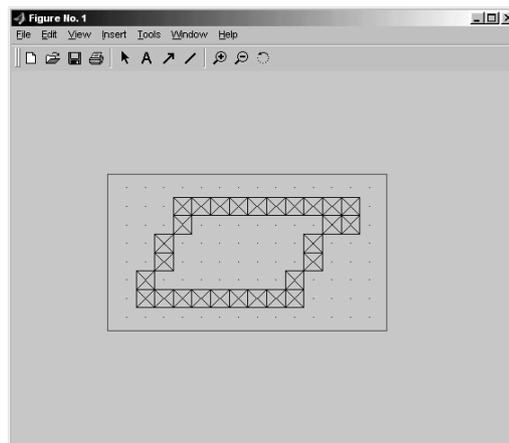
Рис. 1. Динамограммы основных состояний скважинного насоса: а - нормальная работа; б, в — незаполнение насоса; г, д — утечка в приемном клапане; е, ж — утечка в нагнетательном клапане; з — низкая посадка плунжера; и — высокая посадка плунжера.

2. Цифровое представление динамограмм

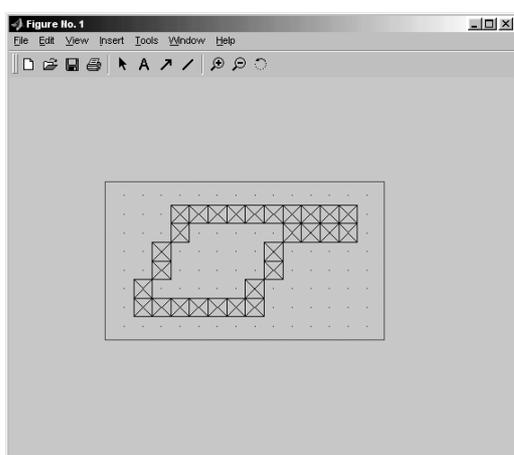
В качестве исходных данных для подачи на вход нейросети используется информация, получаемая от стационарных или переносных средств системы динамометрирования, поэтому на первом этапе разработки системы диагностирования на основе нейронной сети встает задача выбора способа программного представления (оцифровки) динамограмм. Возможны два пути ее решения: задание точек динамограммы действительными числами (в десятичной системе счисления) и двоичными символами (в виде матрицы нулей и единиц). Проанализировав реальные и идеализированные динамограммы и принимая во внимание, что вид идеальной динамограммы зависит только от динамического уровня жидкости в скважине, приходим к выводу, что более универсальным будет использование двоичной системы счисления. К тому же, данные такого типа проще обрабатывать с помощью логических операций. Выделяя характерные элементы изображения девяти основных типовых динамограмм, можно предложить матричную форму цифрового представления динамограмм в бинарном коде с разрешением 14×8 . При необходимости распознавания более сложных динамограмм требуется увеличивать размер матрицы, что влечет за собой усложнение структуры нейросети, затрудняя ее реализацию.



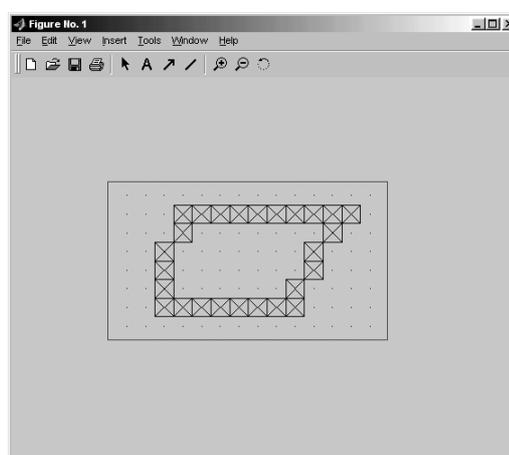
а)



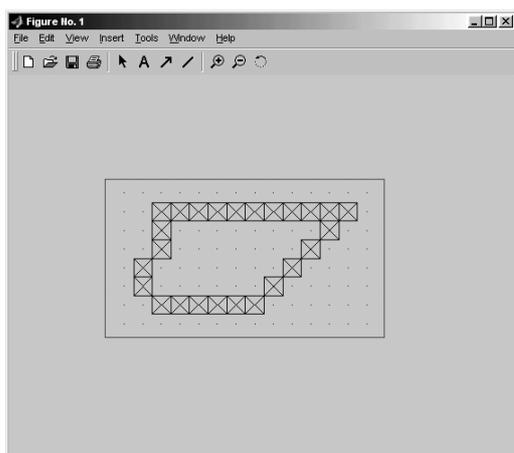
б)



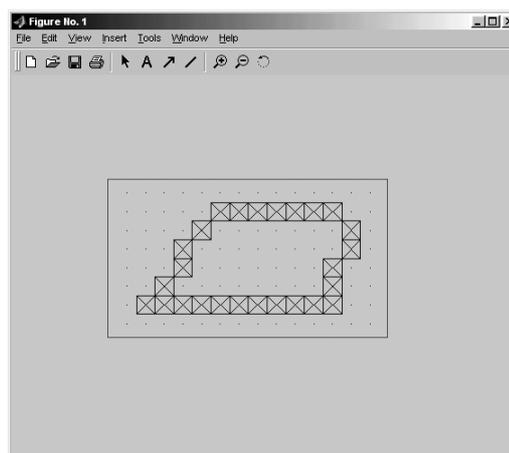
в)



г)



д)



е)

Рис. 3. Окна визуализации векторно-матричного представления динамограмм при работе программы SHOWDIN.M: а - нормальная работа; б, в - незаполнение насоса; г, д - утечка в приемном клапане; е - утечка в нагнетательном клапане.

На вход сети подается предварительно сформированный вектор-строка din_i , характеризующий вид анализируемой динамограммы. На выходе сети получаем численное значение коэффициента принадлежности распознаваемой динамограммы к каждой из девяти типовых динамограмм.

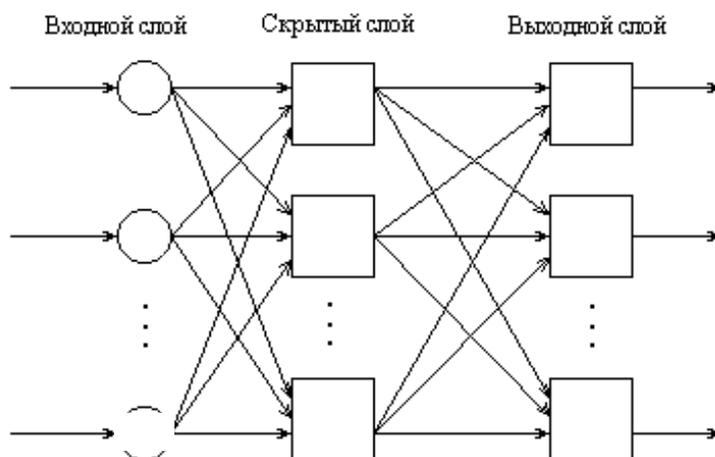


Рис. 5. Структура двухслойной прямонаправленной нейронной сети для анализа динамограмм.

Все исследование и анализ качества работы сети выполнен в приложении Neural Network Toolbox в среде MATLAB 6.1 [2]. Модель системы диагностики представляет собой структуру из пяти файлов-программ, показанную на рис. 6. Фрагменты программы создания и моделирования работы сети NETCREATE.M приведены в Приложении.

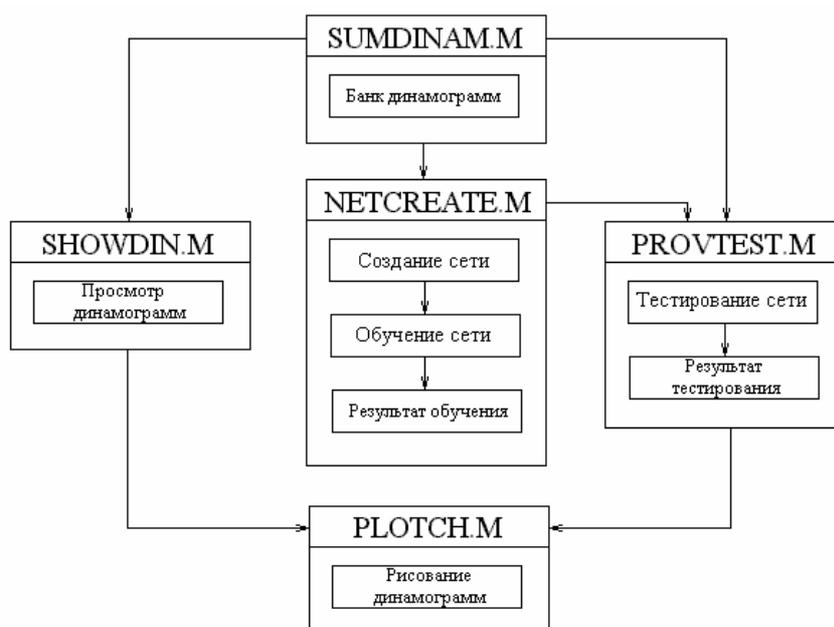


Рис. 6. Структура модели системы диагностики ШГНУ.

Обучение сети проведено итеративным градиентным методом с помощью комбинированного алгоритма обратного распространения ошибки с возмущением и адаптацией параметра скорости настройки. Фрагмент текста программы, задающей параметры обучения в составе файла NETCREATE.M, также приведен в Приложении. При запуске этого файла на экране появляется график, подобный изображенному на рис. 7, отображающий зависимость величины ошибки обучения от числа циклов обучения.

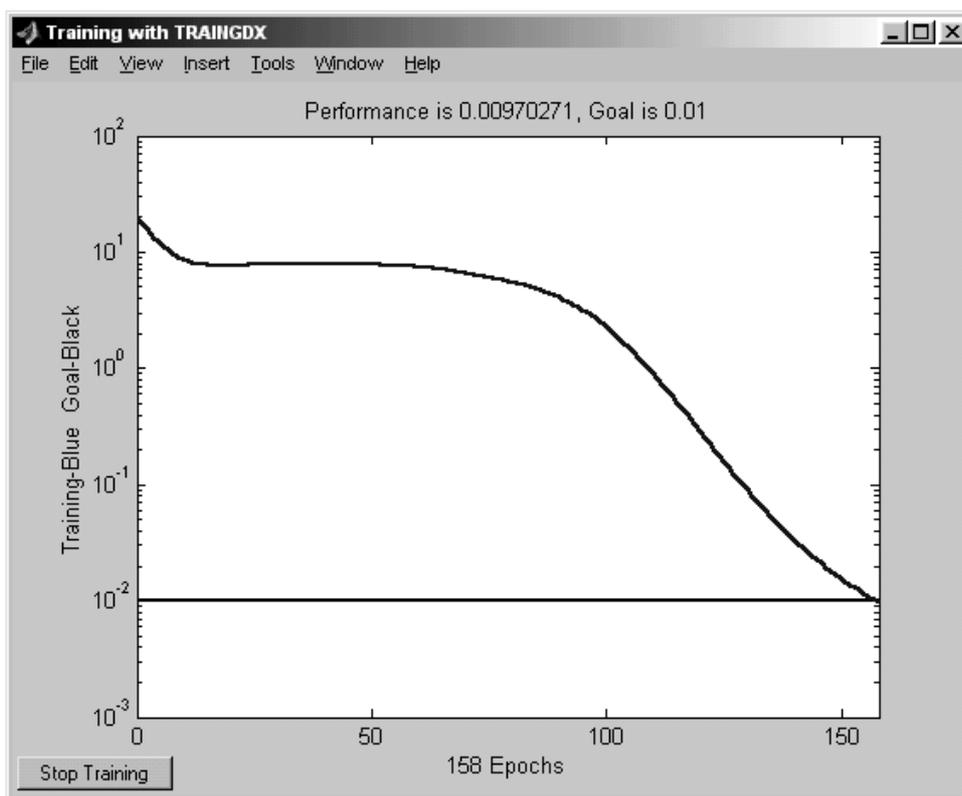


Рис. 7. Зависимость ошибки от числа циклов обучения сети.

4. Тестирование нейронной сети

Тестирование сети, основанное на расчете коэффициента принадлежности распознаваемой динамограммы к типовым, выполняет файл PROVTEST.M, текст которого приведен в Приложении. Если запустить этот файл, например, для анализа динамограммы типа 4, что показана на рис. 3,г (характерной при наличии утечки в приемном клапане), то на экран последовательно будут выводиться изображения идеализированной динамограммы (см. рис. 8), зашумленной динамограммы (см. рис. 9), и результат работы сети в виде гистограммы расчетных значений коэффициента принадлежности при распознавании зашумленной динамограммы (см. рис. 10). Ниже на рис. 11 ... 13 приведены окна, появляющиеся при проверке работы сети на примере распознавания динамограммы типа 9, пока-

занной на рис. 4, и (характерной для высокой посадки плунжера). Способность сети к распознаванию зашумленных динамограмм важна по той причине, что при их снятии на станке-качалки возникает множество помех, обусловленных работой датчиков, аналого-цифровых преобразователей в составе контроллера, а также силами трения, инерционными нагрузками, вынужденными и собственными колебаниями столба жидкости и т. д.

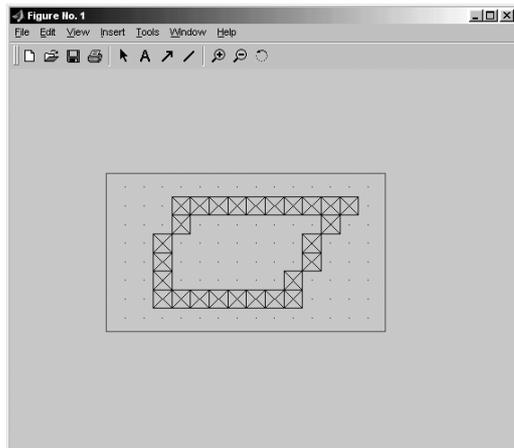


Рис. 8. Окно идеализированной динамограммы 4.

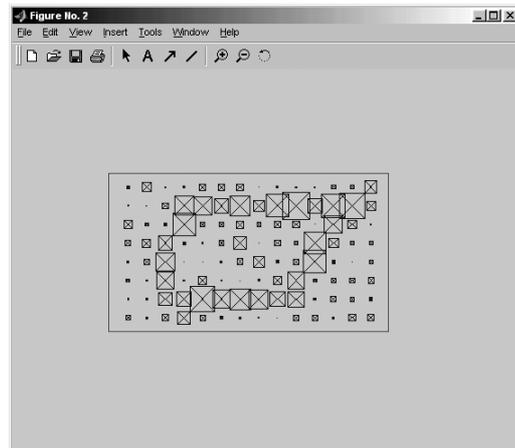


Рис. 9. Окно зашумленной динамограммы 4.

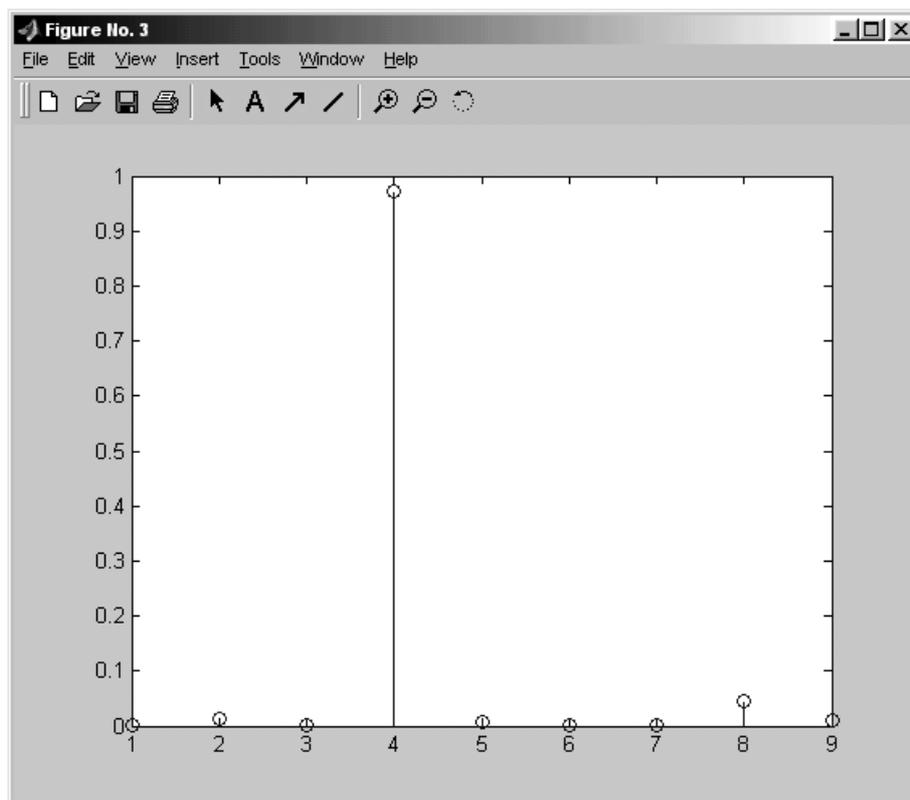


Рис. 10. Окно вывода результатов распознавания динамограммы 4 при уровне зашумления 0,3.

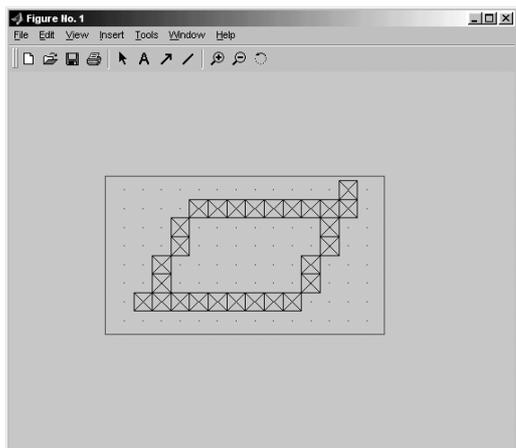


Рис. 11. Окно идеализированной динамограммы 9.

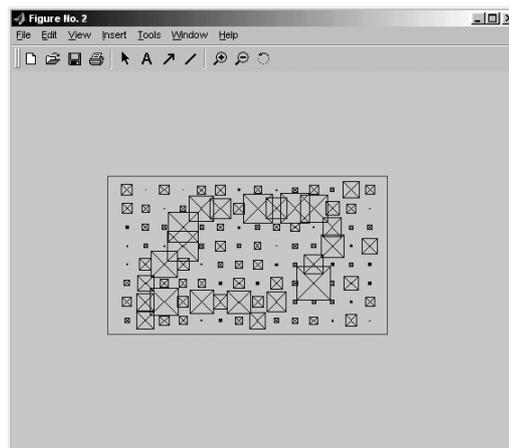


Рис. 12. Окно зашумленной динамограммы 9.

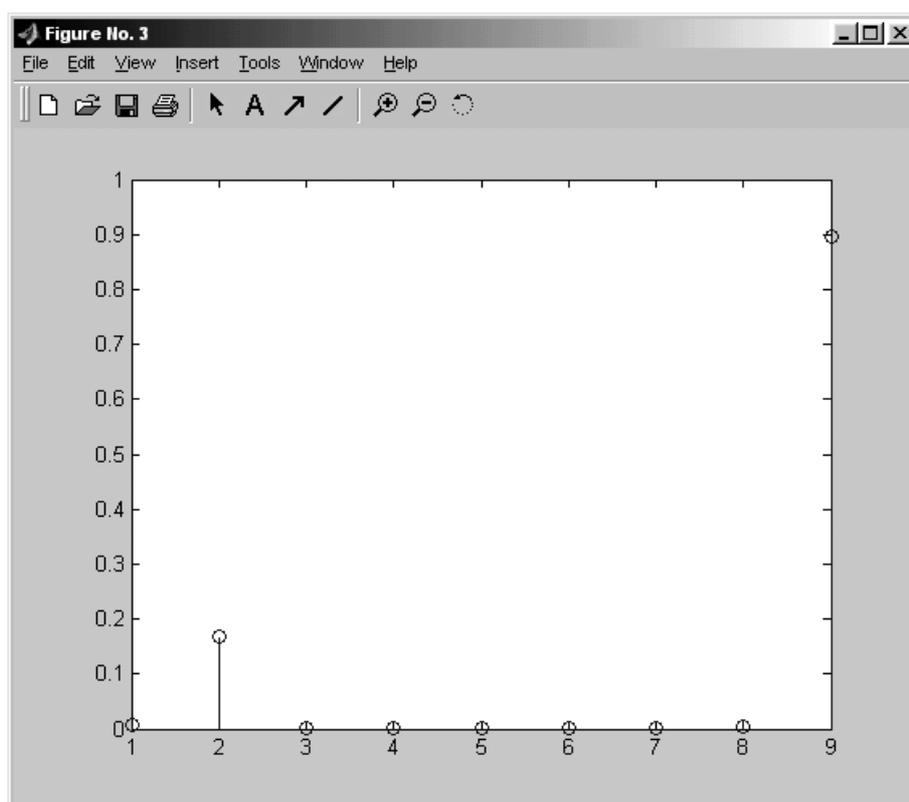


Рис. 13. Окно вывода результатов распознавания динамограммы 9 при уровне зашумления 0,4.

Для проверки способности сети к распознаванию динамограмм при различных уровнях зашумления проведено исследование (на примере динамограмм 4 и 9) зависимости коэффициента принадлежности от уровня зашумления путем проведения трех проб на каждый уровень зашумления. Полученные результаты представлены на рис. 14.

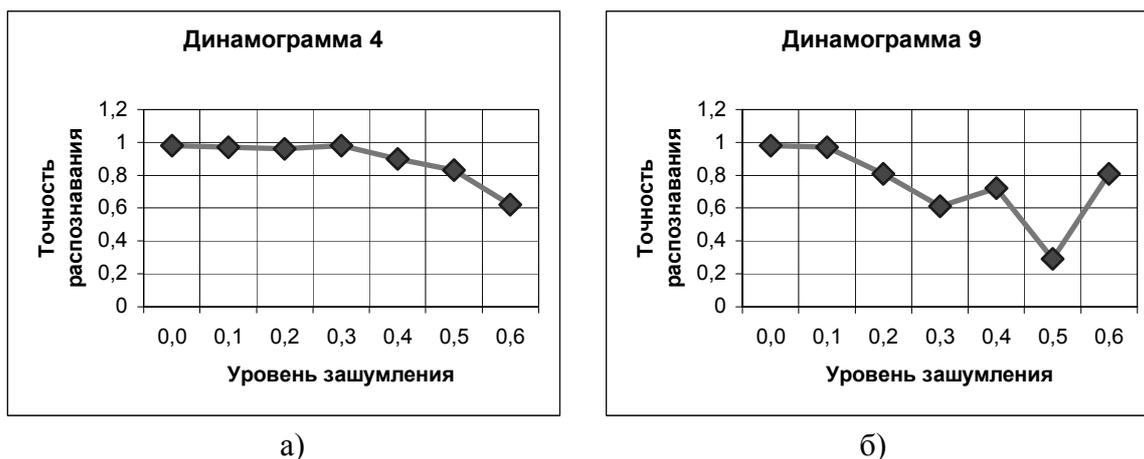


Рис. 14. Зависимость точности распознавания от уровня зашумления:
а - для динамограммы 4; б — для динамограммы 9.

Результаты тестирования качества работы предлагаемой нейронной сети приводят к выводу, что рассмотренная структура вполне способна обеспечить достаточно высокую точность распознавания динамограмм, даже при наличии довольно значительного уровня помех. Преимущество данного решения заключается в общности подхода к анализу всего разнообразия динамограмм и возможности развития системы путем параметрирования выбранной структуры с последующим обучением.

5. Техническая реализация системы диагностики на основе нейронной сети

Возможны несколько вариантов реализации предложенных решений по построению системы диагностики на основе нейронной сети в зависимости от оснащенности ШГНУ средствами контроля, наличия каналов связи, предполагаемого уровня управления и технологии обработки динамограмм.

При реализации системы диагностики на высшем уровне технологического контроля и управления целесообразны следующие варианты ее построения:

- в отсутствии на станке-качалке стационарных средств динамометрирования обращаемся к традиционному решению задачи диагностики на основе данных периодического контроля с использованием переносных средств динамометрирования. Для функционирования системы в этом случае у технологической службы должен иметься персональный компьютер и соответствующее программное обеспечение, например, MATLAB 6.1 Professional Edition, в среде которого реализуется рассмотренная система диагностики;

- при оснащении ШГНУ стационарными средствами динамометрирования и системой управления с каналами связи для передачи данных, централизованная обработка динамограмм может быть выполнена на верхнем уровне управления теми же средствами, что и в предыдущем случае;
- при реализации системы диагностики на кустовом уровне или в составе станции управления ШГНУ, требуется оснащение станка-качалки стационарными средствами динамометрирования и применение в системе управления достаточно мощного контроллера со специальным программным обеспечением, рассчитанным на реализацию нейронных сетей (например, Simatic S7 — 300 Outdoor фирмы Siemens). Контроллер SIMATIC S7-300 Outdoor предназначен для работы в тяжелых промышленных условиях, отличающихся сильным воздействием вибрации и тряски, повышенной влажности, широким диапазоном рабочих температур. На основе программы диагностики и использования этого контроллера возможно создание специальной системы управления электроприводом, использующей для выработки управляющих сигналов данные, полученные при анализе динамограмм.

Литература

1. Алиев Т. М., Тер-Хачатуров А. А. Автоматический контроль и диагностика скважинных штанговых насосных установок.— М.: Недра, 1988.— 232 с.
2. Медведев В. С., Потемкин В. Г. Нейронные сети. MATLAB 6 / Под общ. ред. В.Г. Потемкина.— М.: Диалог–МИФИ, 2002.— 496с.

Приложение

Программа задания динамограмм

```
SUMDINAM.M
function [masdin, targ] = sumdinam ( );
% Файл-функция sumdinam определяет 9 векторов входа,
% каждый из которых содержит 112 элементов. Этот массив называется masdin.
% Файл-функция формирует выходные переменные masdin и targ,
% которые определяют массивы динамограмм и целевых векторов.
% Задание динамограмм
din1 = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ...
0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 ...
0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 ...
0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 ...
0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 ...
0 1 0 0 0 0 0 0 0 0 1 0 0 0 ...
0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 ...
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ];

din2 = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ...
0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 ...
0 0 0 1 0 0 0 0 0 0 0 1 1 0 ...
0 0 1 0 0 0 0 0 0 0 1 0 0 0 ...
```

```
00100000001000 ...
010000000010000 ...
01111111110000 ...
00000000000000];
```

```
din3 = [00000000000000 ...
00011111111110 ...
00010000011110 ...
00100000100000 ...
00100000100000 ...
01000001000000 ...
01111111000000 ...
00000000000000];
```

```
din4 = [00000000000000 ...
00011111111110 ...
00010000000100 ...
00100000001000 ...
00100000001000 ...
00100000010000 ...
00111111110000 ...
00000000000000];
```

```
din5 = [00000000000000 ...
00111111111110 ...
00100000000100 ...
00100000001000 ...
01000000010000 ...
01000000100000 ...
00111111000000 ...
00000000000000];
```

```
din6 = [00000000000000 ...
00000111111110 ...
00001000000010 ...
00010000000010 ...
00010000000100 ...
00100000000100 ...
01111111111110 ...
00000000000000];
```

```
din7 = [00000000000000 ...
00000011111000 ...
00000100000100 ...
00001000000010 ...
00010000000010 ...
00100000000100 ...
01111111111110 ...
00000000000000];
```

```
din8 = [00000000000000 ...
00001111111110 ...
00010000000100 ...
00010000000100 ...
00100000001000 ...
00100000001000 ...
01111111110000 ...
01000000000000];
```

```
din9 = [00000000000010 ...
00001111111110 ...
```

```

0 0 0 1 0 0 0 0 0 0 0 1 0 0 ...
0 0 0 1 0 0 0 0 0 0 0 1 0 0 ...
0 0 1 0 0 0 0 0 0 0 1 0 0 0 ...
0 0 1 0 0 0 0 0 0 0 1 0 0 0 ...
0 1 1 1 1 1 1 1 1 1 0 0 0 0 ...
0 0 0 0 0 0 0 0 0 0 0 0 0 0 ] .

```

```

% Создание массива динамограмм
masdin = [din1,din2,din3,din4,din5,din6,din7,din8,din9];
% Формирование матрицы целевых векторов
targ = eye(9);

```

Программа визуализации динамограмм

```

PLOTCH.M
function plotch(a)
% Файл - функция PLOTCH выводит на экран
% 112 — ти элементный вектор в виде шаблона размерами 14x8
% PLOTCH(A)
% A — 112 — ти элементный вектор, представленный сеткой 14x8.
% Создание прямоугольника, изображающего единичное значение
% элемента матрицы, которой задана динамограмма
x1 = [-0.5 -0.5 +0.5 +0.5 -0.5];
y1 = [-0.5 +0.5 +0.5 -0.5 -0.5];
% Переход к следующему элементу матрицы
x2 = [x1 +0.5 +0.5 -0.5];
y2 = [y1 +0.5 -0.5 +0.5];
% Задание рамки, в которой располагается изображение динамограммы
newplot;
plot(x1*15+6.6,y1*8.5+3,'m');
% Задание осей координат
axis([-2.5 18.5 -2.5 10.5]);
axis('equal')
% Скрытие осей
axis off
% Задание цикла для вывода на экран изображения динамограммы
hold on
for j=1:length(a)
x = rem(j-1,14)+0.14;
y = 6-floor((j-1)/14)+0.5;
plot(x2*a(j)+x,y2*a(j)+y);
end
hold off

```

Для вывода на экран всех идеализированных динамограмм предназначен нижеследующий файл.

```

SHOWDIN.M
% Файл — сценарий для вывода на экран шаблона для
% j - динамограммы
script;
[masdin,targ] = sumdinam;
% Выбор номера динамограммы
j = 8;
% Выбор динамограммы по номеру из массива динамограмм
tj=masdin(:,j);
% Представление динамограммы шаблоном 14x8
din=reshape(tj,14,8)
% Вывод изображения динамограммы на экран
plotch(masdin(:,j));

```

В результате запуска этого файла при изменении j от 1 до 9 на экран будут выводиться все заданные идеализированные динамограммы.

Программа создания модели нейронной сети

```
NETCREATE.M
% Файл — сценарий для создания и обучения
% прямонаправленной нейронной сети
script;
[masdin,targ] = sumdinam;
[R,Q] = size(masdin);
[S2,Q] = size(targ);
% Создание сети прямого направления
% S1- число нейронов первого слоя
% S2 — число нейронов второго слоя
S1 = 26;
% Lr — карта слоев сети
Lr = [S1 S2];
% Insize - минимальное и максимальное возможное значение каждого входа
Insize = minmax(masdin);
% net — структура сети
% {'logsig' 'logsig'}- функции активации нейронов, соответственно
% первого и второго слоев
% 'traingdx' — функция обучения методом градиентного спуска
% с возмущением и адаптацией параметра скорости настройки
net = newff(Insize,Lr,{'logsig' 'logsig'},'traingdx')
% Предварительное масштабирование весов и смещений
% второго слоя для приведения в соответствие случайно
% заданных на первом шаге весов и смещений
net.LW{2,1} = net.LW{2,1}*0,01;
net.b{2} = net.b{2}*0,01.
% Параметры обучения сети
net.performFcn = 'sse'; % Критерий качества обучения — сумма
% квадратов ошибок
net.trainParam.goal = 0.01; % Заданное предельное значение суммы
% квадратов ошибок
net.trainParam.show = 20; % Интервал вывода результатов на экран
net.trainParam.epochs = 5000; % Число циклов обучения
net.trainParam.mc = 0.95; % Параметр возмущения
% Обучение
% P — массив шаблонов
P = masdin;
% T — массив целей
T = targ;
net = train(net,P,T);
```

Программа тестирования нейронной сети

```
PROVTEST.M
% Файл — сценарий для вывода на экран результатов работы сети
script
% Выбор номера динамограммы
j = 1;
% Задание уровня зашумления
noiselevel = 0;
% Формирование зашумленной динамограммы
tj = masdin(:,j)+randn(112,1)*noiselevel;
% Формирование распознанной динамограммы и вывод ее на экран
% в виде отдельного окна
```

```
figure(1);
plotch(masdin(:,j));
% Вывод на экран зашумленной динамограммы в виде
% отдельного окна
figure(2);
plotch(tj);
% Моделирование нейронной сети
Outnw = sim(net,tj);
% Вывод результатов работы сети в виде графика - гистограммы
figure(3);
stem(Outnw);
axis([1 9 0 1])
set(gca,'xtick',[1:1:9])
Lb = char([80:60]);
set(gca,'XTickLabel',{Lb(1) Lb(2) Lb(3) Lb(4) Lb(5) Lb(6) Lb(7) Lb(8) Lb(9)})
```

УДК 553.3.044.001.12 : 519.240

РАССТОЯНИЕ МЕЖДУ ОБЪЕКТАМИ ПРИ ПРОГНОЗЕ ПОЛЕЗНЫХ ИСКОПАЕМЫХ

Локтионов А. А., Аргынова А. Х.,

Физико-технический институт Министерства образования и науки

Республики Казахстан, Алматы,

e-mail: loction@satsun.sci.kz

Лось В. Л., Токарский Э. А.

Академия минеральных ресурсов Республики Казахстан, Алматы,

e-mail: los@mail.ru

1. Постановка задачи

В современных исследовательских проблемах непрерывно увеличиваются размеры и сложность подлежащих анализу данных. В полной мере это относится к важнейшей геологической задаче - прогнозу полезных ископаемых с количественной оценкой прогнозных ресурсов [1]. Исключительно важно при этом использовать методы, обеспечивающие возможность анализа данных без наличия априорной информации и слишком сильных предположений. Невозможно переоценить значение извлечения знаний, скрытых в многомерных корреляционных полях и неклассифицированных первичных данных. Их тщательный анализ может подтвердить или опровергнуть первоначальную гипотезу, наметить новые гипотезы и непосредственно повлиять на принятие решений при изучении как научных проблем, так и прикладных геологоразведочных задач.

Если в прошлом структурно-статистический анализ данных и классификация рассматривались в качестве важного, но все-таки вспомогательного этапа, то в настоящее время происходит переоценка значимости этих подходов. Формируется новое самостоятельное направление исследований — **добыча данных** — **Data Mining**. Современное понятие добычи данных включает в себя последовательность этапов все более детального уровня описания, понимания и анализа данных. Целью такого анализа является **извлечение знаний** — **Discovery Knowledge** — о предмете исследований [2, 3].

Интересная в теоретическом и важная в практическом отношении задача при прогнозировании полезных ископаемых заключается в следующем: как проанализировать, описать и использовать соотношения между пространственно-распределенными характеристиками геологической среды? В частности, эта задача возникает при выявлении закономерностей локализации и размещения рудных объектов, на которых основывается прогноз полезных ископаемых. Из современных теоретических представ-

лений и эмпирических данных известно, что распределение многих геологических характеристик в пространстве крайне неоднородно, нелинейно и нерегулярно. Имеются «пики» очень высоких значений (например, концентраций каких-либо металлов) перемежаемые областями низких значений характеристик; сами участки повышенных и пониженных значений характеристик имеют различные размеры и форму. В такой ситуации эффективность статистических критериев связи снижается.

Не менее важной при прогнозировании полезных ископаемых является задача кластеризации данных. Дело в том, что геологические процессы, сопровождающиеся рудообразованием, должны приводить к определенной структурной перестройке геологической среды и, соответственно, должны проявляться в каких-то ее характеристиках. Не рассматривая собственно физико-химические основы механизмов образования полезных ископаемых, (эта проблема далеко выходит за рамки настоящей статьи), отметим лишь, что эти «застывшие во времени» изменения в многомерном векторе параметров, характеризующих структурно-вещественные особенности геологической среды, могут проявляться в виде соответствующих пространственных кластеров сгущения (или разряжения).

Размерность кластеров, набор компонентов векторов и сами центры этих кластеров принципиально, априорно неизвестны. Следует подчеркнуть, что здесь речь идет не о широко обсуждаемой ограниченности точности измерений, а именно о принципиальном отсутствии знаний.

Для адекватного описания используемых при прогнозировании полезных ископаемых нелинейных многомерных геологических данных необходимо применять такие понятия **расстояния** между геологическими объектами и такие **методы кластеризации**, которые одинаково хорошо будут работать и в **линейных**, и в **нелинейных задачах**. Подход к проблеме «расстояний между объектами» с абстрактных позиций теории множеств удовлетворяет этому условию и позволяет получить эффективные алгоритмы количественного анализа взаимосвязи пространственных переменных. Наилучшее решение задачи многомерной кластеризации при отсутствии априорных знаний достигается методами нейроматематики – методами самоорганизующихся нейронных сетей.

Работа выполнена в среде MATLAB версии 5.21 и свободно распространяемого пакета SOM Toolbox 2.0 [4, 5].

В качестве исходных данных использовался банк информации о средней доле основных типов горных пород и среднем силикатном составе элементарных площадей на территории Казахстана. Эти элементарные площадки соответствуют 1 листу масштаба 1: 100 000; размер ячеек составляет примерно 37x37 км; общее количество ячеек равно 1017. Размерность вектора характеристик равна 31: 20 из них соответствуют доле (в %) основных типов пород, 9 описывают химический состав, 2 дают пространственные координаты x и y ячейки.

2. Расстояния между множествами.

Применение к анализу геологических данных

Понятия «расстояние» между объектами, «мера близости» широко используются в научных исследованиях, прикладных задачах и повседневной жизни. В одних случаях такие определения очень наглядны и связаны с ясными количественными соотношениями. В других случаях при высокой наглядности трудно дать числовую меру соответствующего расстояния. Список дисциплин, в которых исследуются и анализируются взаиморасположения, взаимосвязи, взаимовлияния объектов на пространственных и/или временных последовательностях, в последние годы быстро увеличивается. Особенно актуальна эта задача при изучении сложных пространственных (пространственно-временных) структур с элементами хаотичности, фрактальности, когда трудно сформулировать само понятие расстояния (или корреляции). Именно с такой ситуацией мы сталкиваемся в геологических исследованиях при попытке анализа взаимосвязи двух- и трехмерных полей характеристик геологической среды. В частности, при прогнозе полезных ископаемых это взаимосвязь целевых характеристик (плотности оруденения, концентрации каких-либо металлов и т. д.) с прогнозирующими характеристиками, описывающими вещественные, структурные, геометрические свойства геологической среды. Если такие взаимосвязи устойчивы, обладают достаточной силой (в статистической трактовке) и содержательно интерпретируются, то они называются закономерностями размещения полезных ископаемых и являются геологической основой прогноза.

Введем общее понятие расстояния между двумя множествами, которые при пространственной привязке их элементов могут рассматриваться как два тела (структуры) размерностью $2D$ или $3D$.

Для двух непустых конечных множеств A и B их пересечение $A \cap B$ определяет элементы, принадлежащие и множеству A , и множеству B . Обозначим число элементов, входящих в такое пересечение, символом абсолютной величины $|A \cap B|$. Аналогичным образом $|A \cup B|$ будет давать число элементов, входящих в объединение.

Отношение числа элементов, входящих в пересечение множеств, к числу элементов, входящих в объединение,

$$|A \cap B| / |A \cup B|$$

допускает эвристическую интерпретацию. Это отношение измеряет вероятность того, что элемент, по крайней мере, одного из двух множеств является элементом обоих. Оно, таким образом, дает разумную меру «близости», между рассматриваемыми множествами. Величину

$$DIST(A, B) = 1 - |A \cap B| / |A \cup B|$$

следует рассматривать [6] как меру расстояния (меру «удаленности») между множествами A и B .

Если $A = B$, $DIST(A, B) = 0$.

Если $|A \cap B| = \emptyset$, $DIST(A, B) = 1$

Определенная таким образом величина $DIST(A, B)$ удовлетворяет [6] всем свойствам функции метрики в пространстве, элементами которого являются конечные непустые множества:

$DIST(A, B) + DIST(B, C) > DIST(A, C)$, неравенство треугольника;

$DIST(A, B) = 0$ если и только если $A = B$, положительная определенность;

$DIST(A, B) = DIST(B, A)$, симметрия.

В геологии при прогнозировании полезных ископаемых особый интерес представляет исследование взаимосвязи характеристик, представляющих пространственные $2D$ поля.

Для примера на рис. 1 показаны графики $DIST$, описывающие взаимосвязь полей плотности оруденения $Fe-Au$ и $Pb-Zn$ на свободной от современных рыхлых отложений территории Казахстана. Между полями плотности $Pb-Zn$ в зависимости от относительной концентрации металлов фиксируются небольшие расстояния, которые плавно увеличиваются до $0,4 - 0,5$ только при высоких относительных концентрациях $\approx 0,8$, т. е. в диапазоне самих «вершин» полей плотности Pb и Zn . В то же время величина $DIST$ в паре $Fe-Au$ резко возрастает и достигает 1 , т. е. максимума, уже при относительных концентрациях $\approx 0,5$.

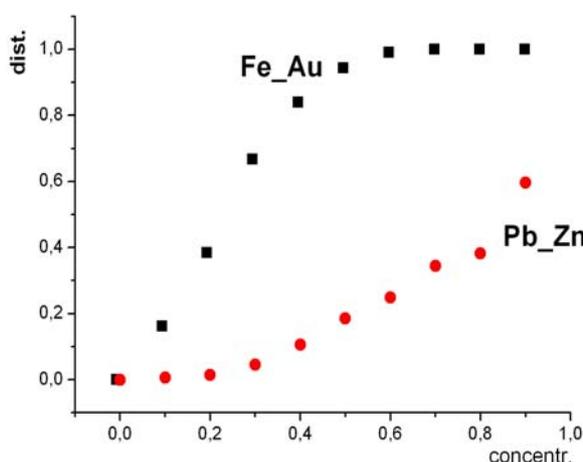


Рис. 1. Анализ расстояний между множествами концентраций $Fe - Au$, $Pb - Zn$.

Для величин относительных концентраций, превышающих пороговые значения $0,1$, $0,3$, $0,6$ соответственно, на рис. 2 приводятся поля плотности оруденения (в логарифмической шкале) для пары металлов $Pb-Zn$, а на рис. 3 — для пары $Fe-Au$ [7].

Как и следовало ожидать, металлы Pb и Zn тесно ассоциируют между собой, а $Fe-Au$ являются «антагонистами».

Анализ расстояний между множествами концентрации Pb - Zn

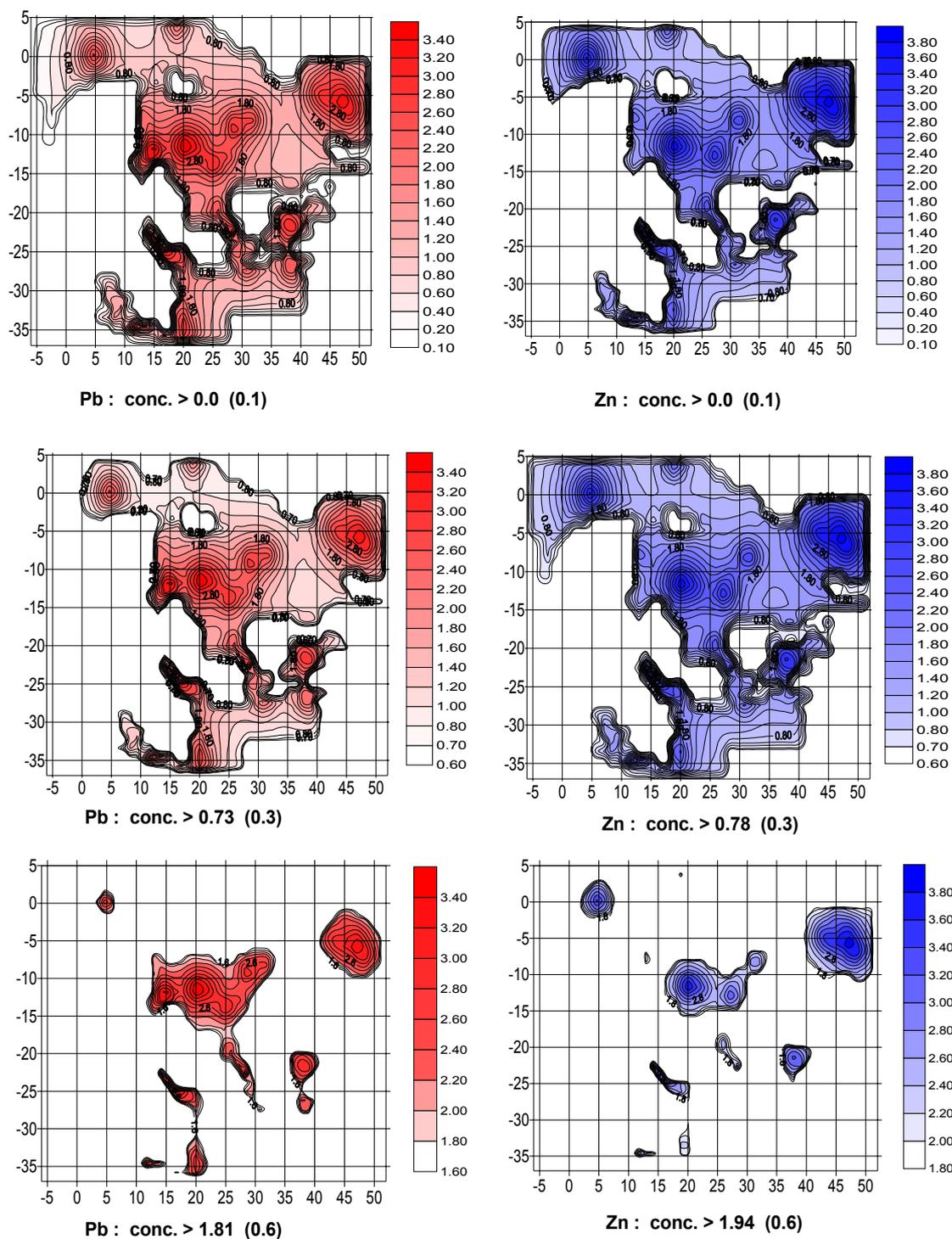


Рис. 2. Геологическая карта концентраций Pb-Zn.

Анализ расстояний между множествами концентраций Au - Fe

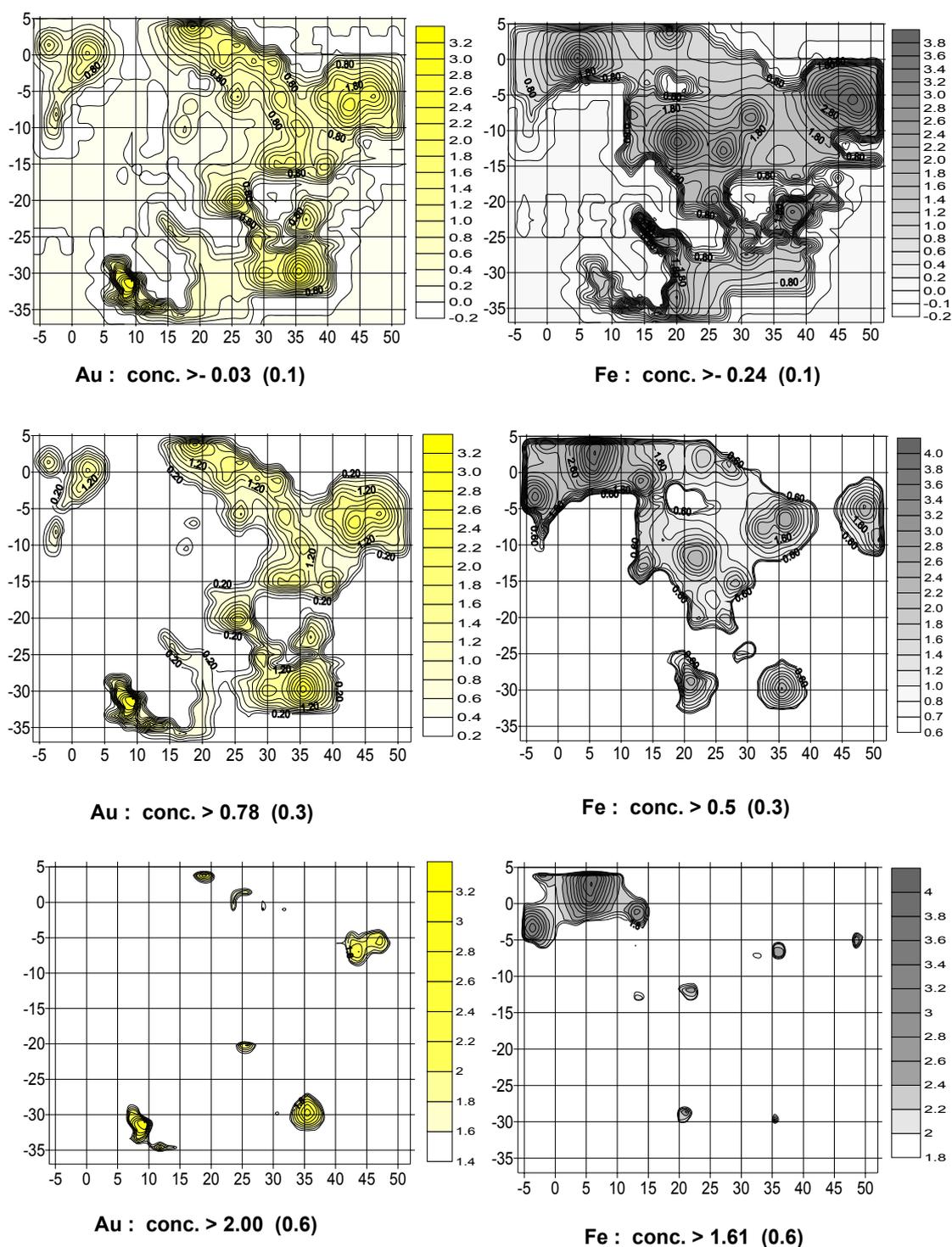


Рис. 3. Геологическая карта концентраций Au-Fe.

Значение *DIST* и ее изменение в зависимости от выбранного порога концентрации позволяет описать структуру взаимоотношений полей распределения характеристик геологической среды. Такой подход достаточно

тесно связан с методами исследования морфологии случайных структур на основе функционалов Минковского.

При необходимости разным значениям полей можно на основе каких-либо содержательных соображений приписать различные «веса». Например, обычно при прогнозе полезных ископаемых нас особенно интересуют области высокой плотности оруденения. Поэтому соответствующим значениям поля при вычислении *DIST* можно придать «вес», пропорциональный их информативности по Шеннону. Обычно вероятность встречи высоких плотностей оруденения мала, а «вес», соответственно, ставится большим.

3. Кластеризация данных

Существуют серьезные основания полагать, что структура экспериментальных данных отражает структуру изучаемых процессов в фазовом пространстве соответствующей размерности. Поэтому понятие «изучение структуры данных» становится почти синонимом понятия «изучение структуры процессов». Классификация помогает найти ключевые абстракции и механизмы, позволяет получить более простую архитектуру изучаемой системы. Классифицируя, мы объединяем в одну группу объекты, имеющие одинаковое строение или одинаковое поведение. Основу классификации в отсутствии априорных знаний составляет кластерный анализ [8, 9].

Кластерные алгоритмы, в принципе, обеспечивают количественный анализ выделения групп, находящихся в исходных данных. Следует, однако, подчеркнуть, что решение задач кластеризации является исключительно сложным процессом [10–12]: в реальной ситуации кластеры редко бывают компактными и хорошо разделенными друг от друга. Поэтому вместо одного единственного алгоритма разделения данных на составляющие кластеры чаще используется итеративная последовательность итеративных алгоритмов. В настоящей работе последовательно использованы алгоритм самоорганизующихся карт Кохонена и алгоритм К-средних. SOM-анализ [13, 14] переводит большую совокупность многомерных данных (тысячи, десятки тысяч значений) в карту с несколькими десяткам (сотнями) ячеек. Алгоритм К-средних [15, 16] выполняет кластерный анализ на этой сетке ячеек SOM-карты.

Сеть Кохонена распознает многомерные кластеры в данных, оценивает близость классов. Упорядочение **многомерных входных** векторов в виде **двухмерной выходной** карты выражается в том, что чем ближе координаты двух векторов на карте, тем ближе они и в пространстве входов (но не наоборот!) Таким образом, исследователь может улучшить свое понимание структуры данных — выполнить разведочный анализ данных. Сети Кохонена можно использовать и в тех задачах классификации, где классы

уже заданы, тогда преимущество будет в том, что сеть сможет выявить сходство между различными классами.

Основное назначение SOM-карт заключается в преобразовании и отображении многомерных входных данных в виде (обычно) двухмерного массива выходных данных: выходной или топологической карты Кохонена. Каждый элемент выходной карты — нейрон i описывается вектором весов $\mathbf{w} = [w_{i1}, w_{i2}, \dots, w_{id}]^T$, имеющим ту же размерность d , что и размерность входных данных. Число нейронов в топологическом слое определяется эвристически.

На стадии инициализации весам всех нейронов присваиваются случайные значения в интервале $0.0-1.0$. Перед началом работы все компоненты вектора входных данных также нормируются на 1.0 .

Принципиальным для процедуры SOM анализа является соревновательный принцип нахождения нейрона с набором весов, ближайшим к набору компонент выбранного вектора \mathbf{x} . В качестве меры расстояния $\|\cdot\|$ обычно применяется евклидово расстояние:

$$\|\mathbf{x} - \mathbf{w}_c\| = \min_i \{\|\mathbf{x} - \mathbf{w}_i\|\}$$

по минимуму которого и выбирается нейрон-победитель \mathbf{w}_c .

Второй принципиальный элемент алгоритма состоит в обучении не только нейрона-победителя, но и его «соседей», хотя и с меньшей скоростью; для этого нейроны выходного слоя упорядочиваются, образуя двухмерные решетки, в которых положение нейронов маркируется вектором \mathbf{r} :

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \eta(t) A_{ci}(t) [\mathbf{x}(t) - \mathbf{w}_i(t)]$$

$$A_{ci}(t) = A(\|\mathbf{r}_c - \mathbf{r}_i\|, t)$$

Здесь t задает шаг обучения, $\mathbf{x}(t)$ — случайно выбранный входной вектор на шаге t , $\eta(t)$ — темп обучения, $A_{ci}(t)$ — функция соседства. При гауссовской форме функции соседства она равна единице для нейрона-победителя с индексом c и экспоненциально уменьшается с расстоянием. Как темп обучения $\eta(t)$, так и радиус взаимодействия нейронов $A_{ci}(t)$ постепенно уменьшаются в процессе обучения, так что на конечной стадии обучения адаптируются веса только нейронов-победителей.

Результаты SOM анализа трудно интерпретировать. На рубеже 90-х годов для наглядной визуализации результатов работы карт Кохонена был предложен метод U -матрицы — унифицированной матрицы расстояний [17]. Для этого в U -матрице между соседними ячейками топологической карты «вставляются» ячейки-анализаторы. Рассчитывая расстояния между соседними ячейками карты Кохонена на основе использования той же самой метрики, новые величины можно занести в добавленные ячейки-анализаторы. Большие значения будут говорить о том, что веса данного нейрона сильно отличаются от окружающих. Таким образом, полосы из

ячеек с большими значениями будут соответствовать границам между кластерами, а области ячеек с низкими значениями — самим кластерам.

В работе последовательная кластеризация методом SOM и К-средних была выполнена на основе SOM Toolbox 2.0 [4] в среде MATLAB версии 5.21. За центры классов брались результаты SOM-метода. Проводя классифицирование методом К-средних с различным заданным числом центров, по критерию (DB) Девиса — Боулдина [17] можно

$$DB = \frac{1}{c} \sum_{l \neq k} \max \left\{ \frac{S_c(Q_k) + S_c(Q_l)}{d(Q_{kl})} \right\}$$

выбрать наилучший вариант, в котором значение этого критерия будет минимизировано. Здесь c — число классов, d — расстояние между классами k и l ; S_c — внутриклассовое расстояние (стандарт).

В качестве примера приложения такого подхода была рассмотрена кластеризация основных типов горных пород на свободной от современных рыхлых отложений территории Казахстана. Для 1017 векторов данных построенная SOM-карта и график изменения критерия DB при кластеризации методом К-средних показаны на рис. 4. Как видим, оптимум кластеризации достигается при выделении 17–19 классов. Нами принято разбиение на 18 классов.

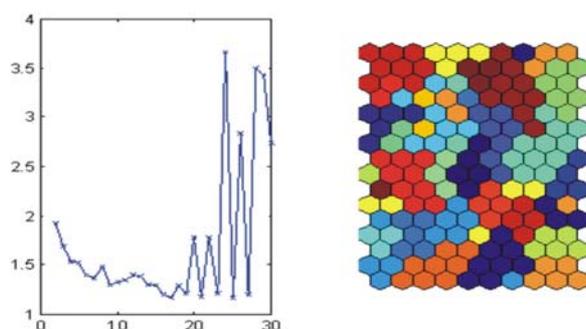


Рис. 4. Справа — SOM-карта, полученная методом анализа нейросети Кохонена по вектору пород из 29 компонент. Слева — график критерия DB в зависимости от числа кластеров.

На рис. 5 показана схема районирования территории Казахстана, построенная на основе выполненной кластеризации. Цифры (с шагом 2, т. е. от 2 до 36) на этой карте определяют номера классов.

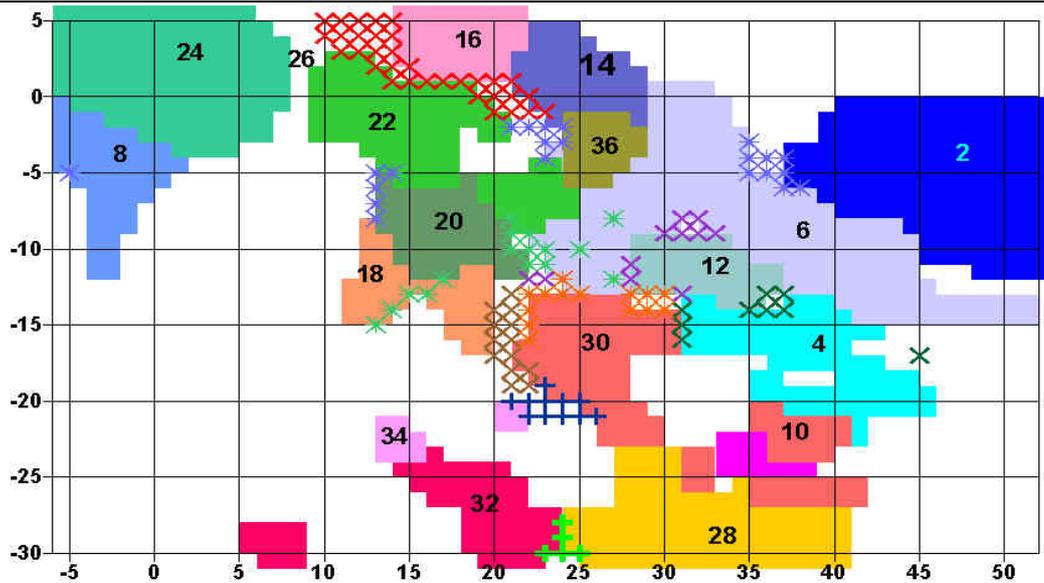


Рис. 5 Районирование территории Казахстана методом SOM по характеристикам геологической среды и пространственным координатам (пространственные координаты - 2 характеристики; доли основных типов горных пород в % - 20 характеристик; силикатный состав среды в % - 9 характеристик).

Примечание. Крестом выделены межкластерные зоны.
Цифрами (с шагом 2) обозначены сами кластеры.

В целом районы соответствуют общему структурному плану геологического строения территории Казахстана. Довольно четко выделяются районы развития карбонатов (Каратау), кислых вулканитов (Центрально-Казахстанский пояс), кислых вулcano-плутогенных образований (Прибалхашье и Шу-Илийский пояс) и т. д.

Следует отметить, что приведенная схема — это только первый шаг. Для реального районирования, особенно если его конечной задачей является прогноз полезных ископаемых, необходимы введение целевых установок районирования, выбор характеристик, наиболее сильно связанных с оруденением прогнозируемого типа (или типов), проверка устойчивости кластеров «шевелением» характеристик, введение или удаление пространственных координат. В частности, на схеме районирования территории Казахстана на рис. 5 определенные сомнения вызывает район 2 (восточная часть Казахстана), в котором объединились достаточно различные структуры (Рудный Алтай, Калба-Нарымский район). Районирование без учета пространственных координат x , y , которые «стягивают» этот кластер, показало, что он распался на структуры, более соответствующие геологическому строению. Карта с новым результатом анализа для этого района приведена на рис. 6.

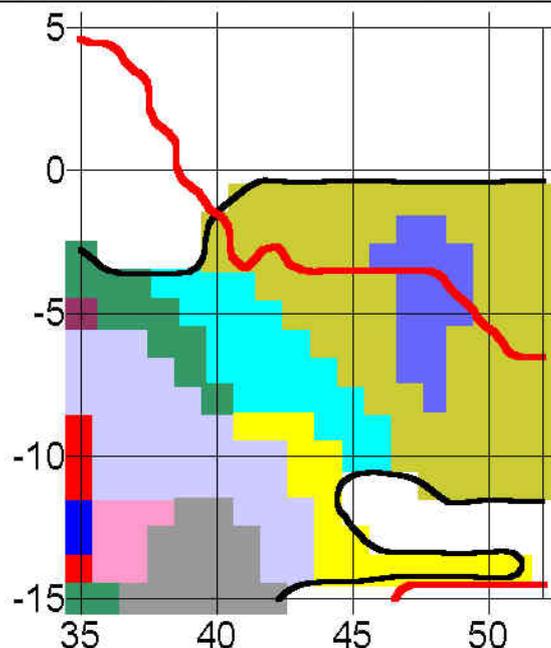


Рис. 6 Районирование территории Восточного Казахстана методом SOM с использованием характеристик геологической среды без пространственных координат (доли основных типов пород в % - 20 характеристик; силикатный состав среды в % - 9 характеристик).

В заключение еще раз подчеркнем, что именно количественное описание и анализ геологических (в частности, рудообразующих) систем как систем многомерных, нелинейных и обладающих определенной способностью к самоорганизации, непрерывное совершенствование адекватных технологий прогноза поведения этих систем могут принципиально повысить уровень качества прогноза полезных ископаемых, обеспечив необходимую надежность оценки недр.

Литература

1. Лось В. Л. Теоретические, методические и технологические основы прогноза рудных объектов // Минералогия Казахстана. Комитет геологии Министерства энергетики и природных ресурсов Республики Казахстан.— 2004.— С.1–15.
2. Introduction to Data Mining and Knowledge Discovery, 3-rd edition.— Two Crows Corp., 1999. (www.twocrows.com).
3. Shearer C. The CRISP-DM Model: The New Blueprint for Data mining // Journal of Data Warehousing.— V.5.— 2000.— N.4.— P.13–23.
4. <http://www.cis.hut.fi/projects/somtoolbox>.
5. Vensanto J., Himberg J., Alhoniemi E., Parhankangas J. SOM Toolbox for Matlab 5.— SOM Toolbox Team, Helsinki University of Technology, 2000.— P.1–60.

6. *Levandowsky M., Winter D.* Distance between Sets // *Nature*.— V.234.— 1971.— P.34–35.
7. *Лось В. Л., Гольдберг И. С.* Рудообразование как самоорганизованный процесс перераспределения металлов // *Геология и минерагения Казахстана*. Алматы.— 2000.— С.116–129.
8. *Everitt. B. S.* Cluster analysis, Wiley.— New York, 1993.
9. *Kaufman L., Rousseeuw P.* Finding Groups in Data: An Introduction to Cluster Analysis.— Wiley: New York, 1990.
10. *Su M. S., Claris N. De, Liu T. K.* Application of neural networks in cluster analysis // *Proc. of the 1997 IEEE Intern. Conf. on Systems, Man and Cybernetics*.— 1997.— P.1–6.
11. *Kothari R., Pitts D.* On finding the number of cluster // *Pattern Recognition Letters*.— V.20.— 1999.— P.405–416.
12. *Hardy A.* On the number of clusters // *Computational statistics and Data Analysis*.— N.23.— 1996.— P.83–96.
13. *Kohonen T.* Self-Organizing Maps. Vol. 30 of Springer Series in Information Sciences.— Springer, 3-rd edition, 1995.
14. *Vensanto J., Alhoniemi E.* Clustering of the Self-Organizing Maps // *IEEE Transactions on Neural Networks*.— 11(2) .— March 2000.— P.586–600.
15. *Han J., Kamber M., Kaufman M.* Data mining, 2001 // (www.cs.sfu.ca/~han/dmbook).
16. www.ucl.ac.uk/MicroCore/HTMLresource/UnsupervisedClustering.htm
17. *Ultisch A., Siemon P.* Kohonen's Self Organizing Feature Maps for Exploratory Data Analysis // *Proc. Intern. Neural Network Conf. (INNC'90)* .— Dordrecht. Netherlands,1990. Kluwer.— P.305–308.
18. *Davies D. L., Bouldin D. W.* A cluster separation measure // *IEEE Trans. Pattern Anal. Machine Intell.* V. POMI.— 1. Apr. 1979.— P.224–227.

УДК 519.711, 621.878:629.11

ИДЕНТИФИКАЦИЯ СТРОИТЕЛЬНЫХ МАШИН КАК НЕЛИНЕЙНЫХ ДИНАМИЧЕСКИХ СИСТЕМ НА ОСНОВЕ НЕЙРОСЕТЕВЫХ ТЕХНОЛОГИЙ

Мещеряков В. А.

*Сибирская государственная автомобильно-дорожная академия (СибАДИ), Омск,
e-mail: vit_mes@sibadi.omsk.ru*

Повышенный интерес к искусственным нейронным сетям со стороны разработчиков систем автоматического управления прежде всего обусловлен способностью нейросетей к адаптации и самообучению, а также возможностью моделирования существенных нелинейностей в структуре сложных динамических объектов управления [1]. При математическом моделировании сложных технических объектов система инженерных и научных расчетов MATLAB с пакетами расширения является наилучшей базой для программной реализации нейросетевых моделей.

Автоматизация строительных машин требует создания их математических моделей, учитывающих динамику рабочих процессов в условиях неполной информации об объекте управления. Нелинейные зависимости между показателями рабочих процессов затрудняют выбор структуры модели и определение ее параметров. Для таких сложных динамических систем целесообразно применение нейросетевых технологий и представление модели в виде динамической нейронной сети. В этом случае идентификация машины заключается в разработке нейросетевой модели на основе входных и выходных экспериментальных данных. В работе приведена методика построения математической модели лабораторной установки, предназначенной для исследования процесса копания грунта землеройной машиной (рис. 1). Нейросетевая модель описывает зависимость силы сопротивления перемещению тележки $F(t)$ от скорости перемещения тележки $v(t)$ и глубины копания $h(t)$. Вектор входных воздействий $u(t) = \{v(t); h(t)\}$ подается на входы исследуемой системы и ее нейросетевой динамической модели (рис. 2). Разница между выходом реального объекта $y^*(t) = F(t)$ (т. е. экспериментально измеренным значением силы) и выходом модели $y(t)$ — ошибка $e(t)$ — используется для настройки параметров математической модели (обучения нейронной сети).

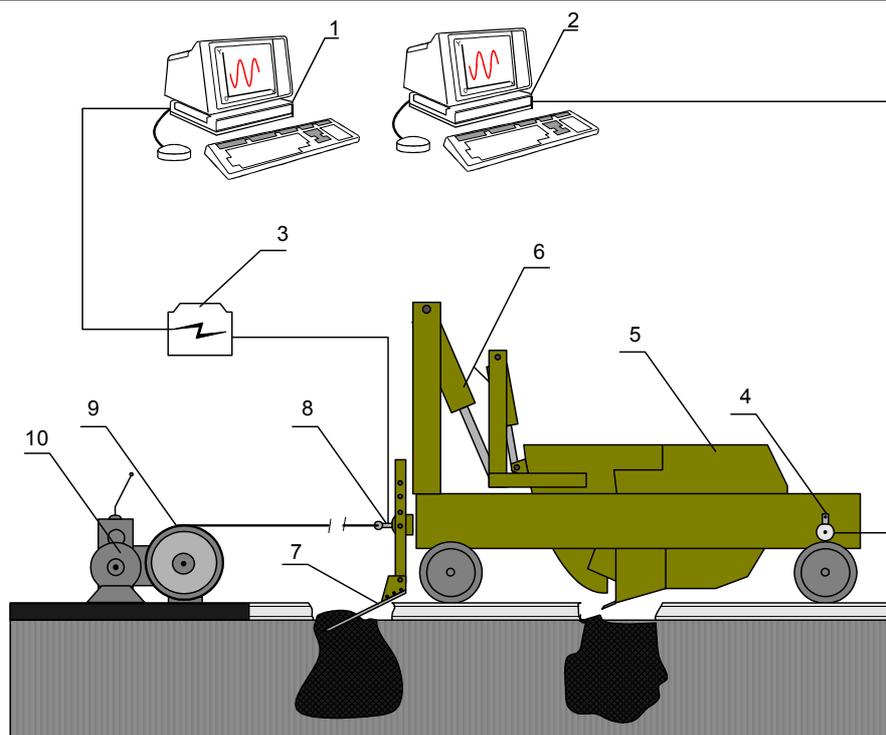


Рис. 1. Экспериментальная установка лаборатории СибАДИ «Грунтовый канал»: 1, 2 — ЭВМ с платами АЦП для обработки сигналов тензодатчика и датчика скорости; 3 — усилитель; 4 — датчик скорости; 5 — тензометрическая тележка на рельсовом ходу; 6 — гидроцилиндры; 7 — прямой нож; 8 — тензозвено; 9 — лебедка канатного привода тележки; 10 — электродвигатель.

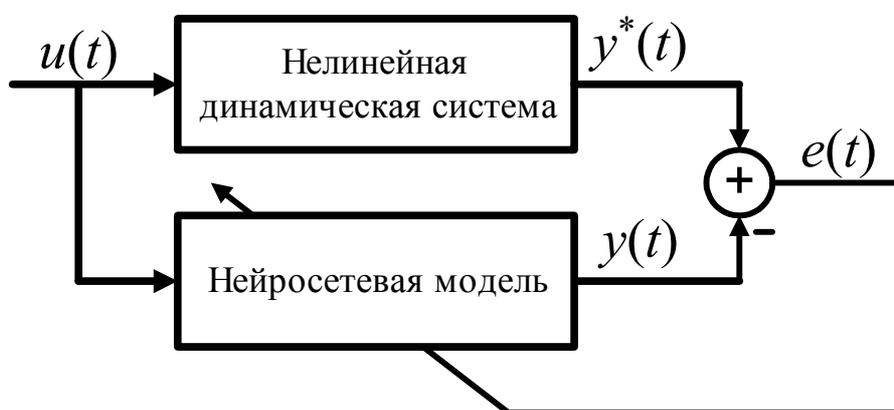


Рис. 2. Идентификация нелинейной динамической системы.

Первый этап создания модели предполагает формирование массивов исходных данных. Разработанный программно-аппаратный комплекс сбора экспериментальных данных включает в себя 2 ЭВМ с платами аналогово-цифровых преобразователей (АЦП) L-Card L-305. Тензозвено предназначено для измерения силы сопротивления копанью грунта. Датчик скорости представляет собой «пятое колесо» с герконом и вращающимися магнитами и предназначен для измерения перемещения и скорости тележ-

ки. Программное обеспечение комплекса разработано в среде Borland Delphi, также использована программа PowerGraph 2.0. Высотные координаты профиля поверхности грунта до и после каждого прохода тележки измерены вручную. Измеренные сигналы с тензозвена и датчика скорости экспортированы для дальнейшей обработки в MATLAB (рис. 3–5). По передним фронтам импульсов с датчика скорости численным дифференцированием в MATLAB рассчитана скорость тележки (рис. 4). Координаты профиля преобразованы в величину глубины копания (рис. 5). С помощью одномерной табличной интерполяции в MATLAB получена зависимость глубины копания от времени (рис. 6). Для исследования выбран интервал времени 15–65 с, на котором наблюдается установившийся процесс.

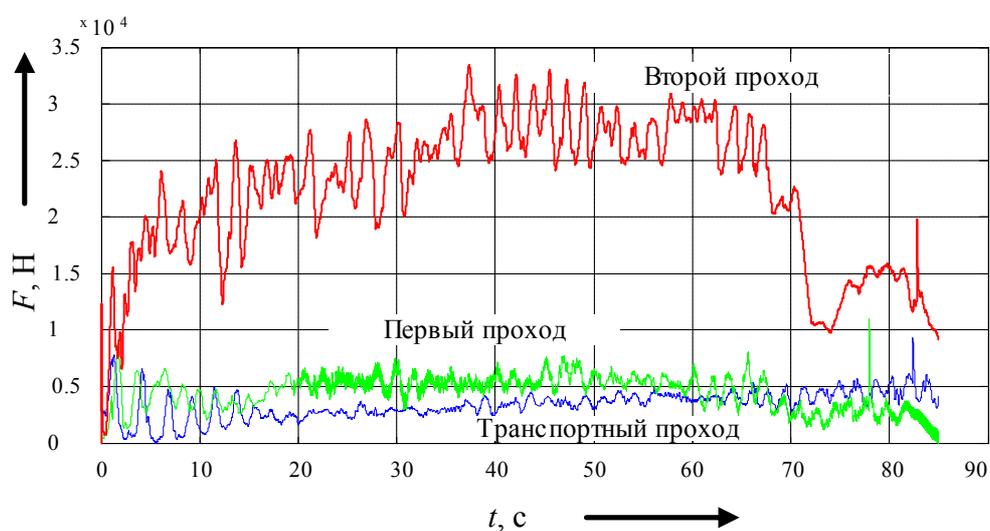


Рис. 3. Зависимость силы сопротивления от времени.

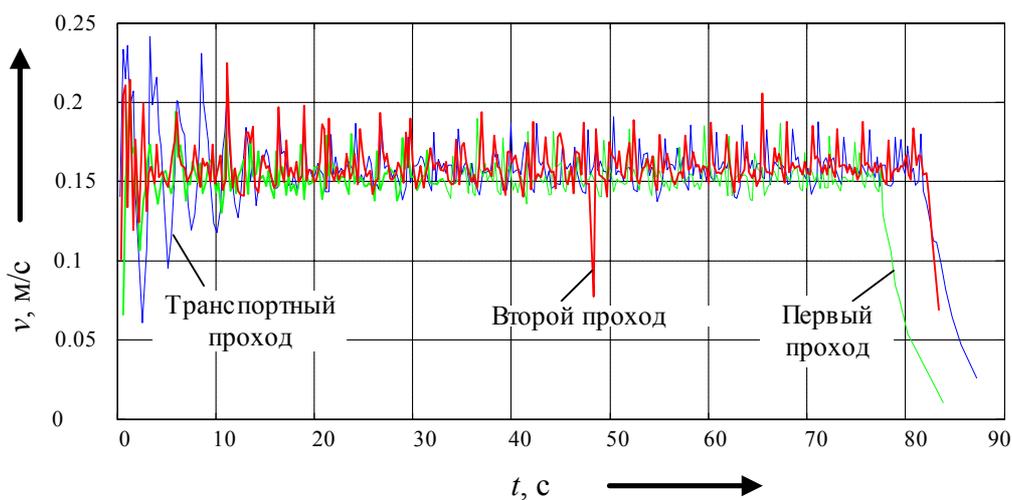


Рис. 4. Зависимость скорости тележки от времени.

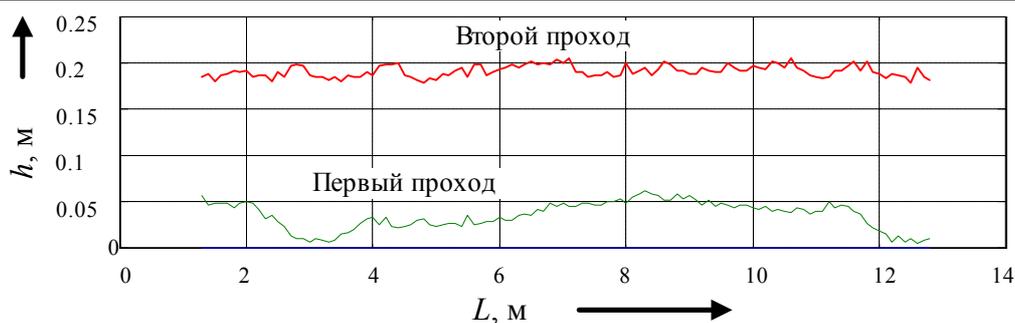


Рис. 5. Зависимость глубины копания от перемещения.

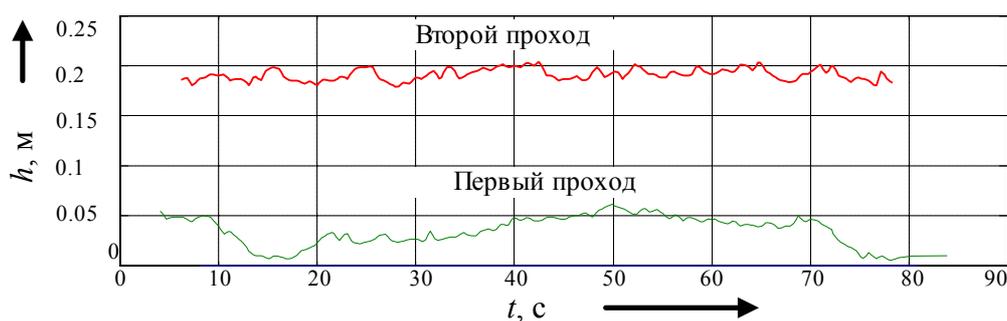


Рис. 6. Зависимость глубины копания от времени.

На втором этапе создания модели необходимо обосновать ее структуру. В настоящей работе приведены результаты моделирования как с помощью нерекurrentной нейронной сети прямого распространения сигнала (программная реализация в MATLAB + Neural Network Toolbox), так и рекуррентной динамической сети (MATLAB + Neural Network Based System Identification Toolbox). Структура нейросетевой модели без обратных связей выглядит следующим образом.

Входами модели являются скорость тележки $v(t)$ и глубина копания $h(t)$. Выход модели — сила сопротивления перемещению тележки $F(t)$. Ярво выраженная периодичность колебаний $v(t)$ обусловлена свойствами канатного привода тележки. Амплитуда $F(t)$ зависит от глубины копания $h(t)$. Частотные характеристики силы $F(t)$ связаны со спектром скорости $v(t)$. В связи с этим нейросетевая модель должна реализовать функцию $F = f(v, h)$.

Для решения такой задачи выбрана двухслойная нейронная сеть прямой передачи сигнала (рис. 7). Отсутствие обратных связей гарантирует устойчивость модели. Сеть имеет один скрытый слой нейронов с нелинейными функциями активации, что делает возможной аппроксимацию нелинейных зависимостей. Выходной слой содержит 1 нейрон с линейной функцией активации, необходимой для экстраполяции зависимостей [2]. Для моделирования динамической системы на входах сети установлены линии задержки TDL [1, 3].

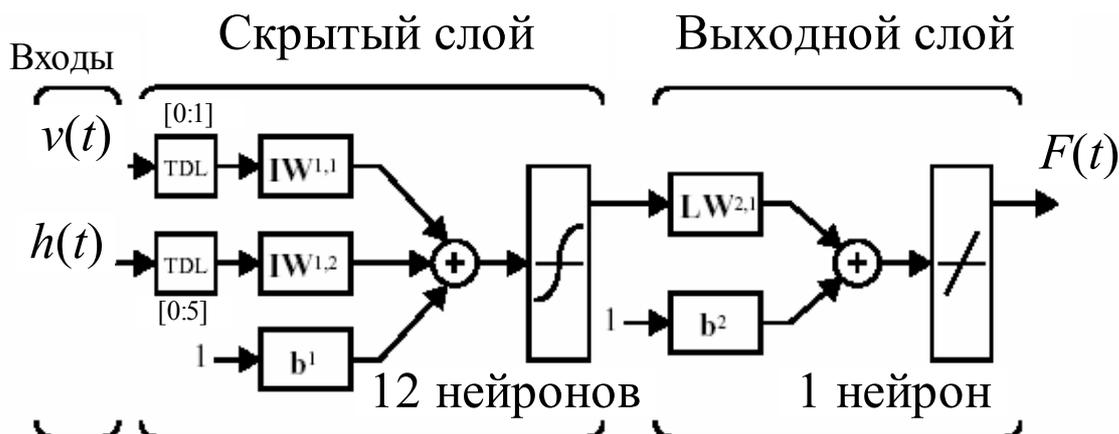


Рис. 7. Структура рекуррентной нейросетевой модели строительной машины.

Третий этап построения модели — подготовка входных и выходных сигналов. Для того, чтобы динамическая нейросетевая модель имела минимальное количество входов, следует обосновать частоту дискретизации сигналов и определить необходимое количество задержанных входов. Для этих целей предлагается методика, основанная на цифровой обработке сигналов.

Проведен спектральный и корреляционный анализ входных и выходных сигналов с использованием пакета расширения MATLAB Signal Processing Toolbox. По спектрограммам и графикам спектральной плотности выбрана частота дискретизации 10 Гц для всех сигналов. При изменении частоты дискретизации выполнена низкочастотная фильтрация сигналов с целью устранения «ложных частот» [4]. Частота дискретизации силы $F(t)$ снижена в 20 раз (с 200 до 10 Гц), частота дискретизации глубины копания $h(t)$ и скорости $v(t)$ повышена примерно в 3 раза. При низкочастотной фильтрации применен дискретный рекурсивный фильтр, имеющийся в составе MATLAB Signal Processing Toolbox. Результат устранения «ложных» высоких частот приведен на рис. 8.

По графикам корреляционных функций входных сигналов $v(t)$ и $h(t)$ определены длины линий задержки на входе модели. Временной интервал, на котором коэффициент автокорреляции сигналов превышает 0,8, составляет 0,1 с для скорости и 0,5 с для глубины копания (рис. 9). Длина линий задержки входов составляет 1 и 5 соответственно. Согласно теореме Колмогорова, достаточное число нейронов в скрытом слое не превышает $2N + 1$, где N — количество задержанных входов [3]. Количество нейронов в скрытом слое принято равным 12 (рис. 7).

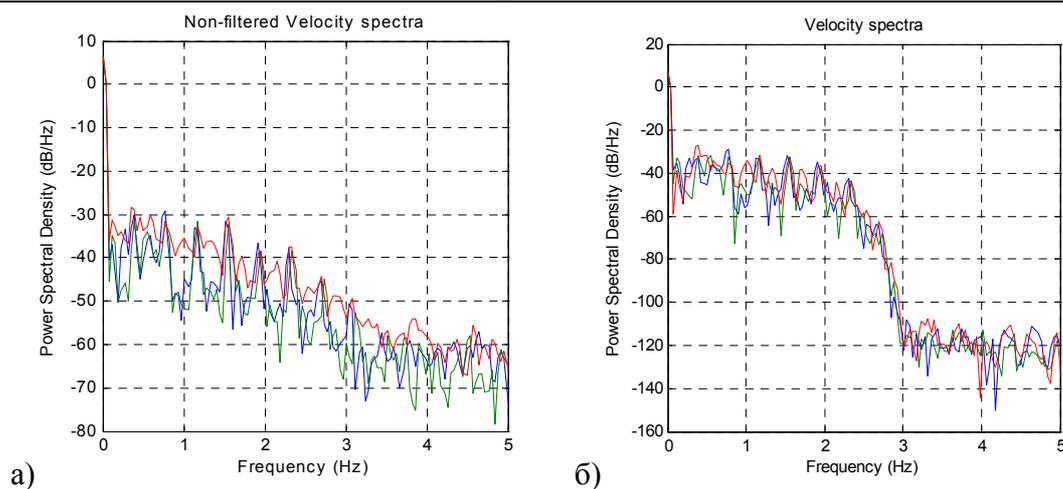


Рис. 8. Спектральная плотность мощности сигнала $v(t)$:
а) до фильтрации; б) после фильтрации «ложных частот».

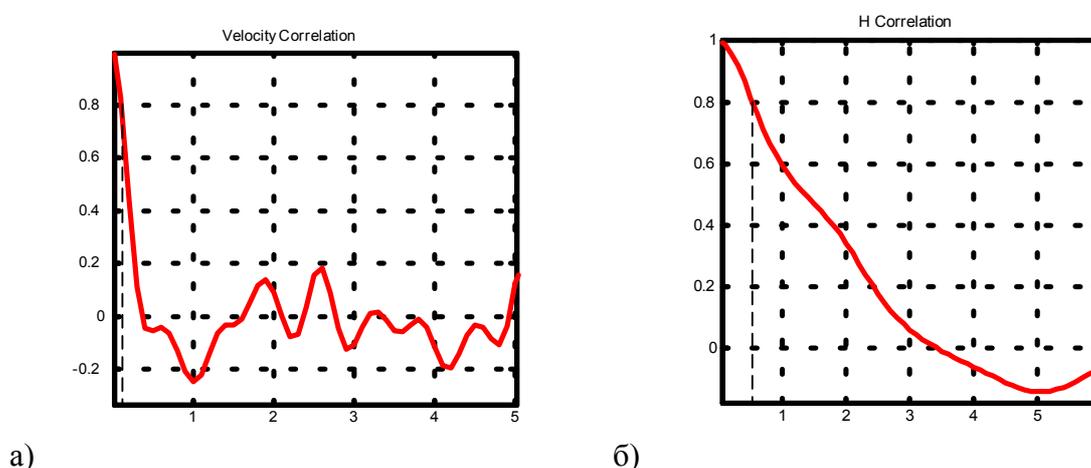


Рис. 9. Автокорреляционные функции входных сигналов:
а) скорости $v(t)$; б) глубины копания $h(t)$.

Для программной реализации нейросетевой модели использован пакет MATLAB Neural Network Toolbox. Поскольку в его составе нет готовой функции для моделирования сети выбранной структуры, применен конструктор класса сети «network» [5]. Визуализация структуры сети с помощью блоков Simulink приведена на рис. 10. Для обучения сети применен градиентный алгоритм — метод Левенберга-Марквардта (**'trainlm'**). За критерий оценки качества обучения принят средний квадрат ошибки (**'mse'**). Начальные значения весов и смещений выбраны случайным образом. Данная нейронная сеть представляет собой нерекурсивный нелинейный фильтр.

Результаты обучения нейросети и моделирования показывают, что после 800 циклов обучения модель верно воспроизводит характер изменения силы $F(t)$ в зависимости от $v(t)$ и $h(t)$ (рис. 11). Отклонения прогнозируемых значений $F(t)$ от фактических обусловлены прежде всего отсут-

ствием обратных связей. Одним из возможных решений этой проблемы является использование рекуррентных нейронных сетей [1, 3, 5].

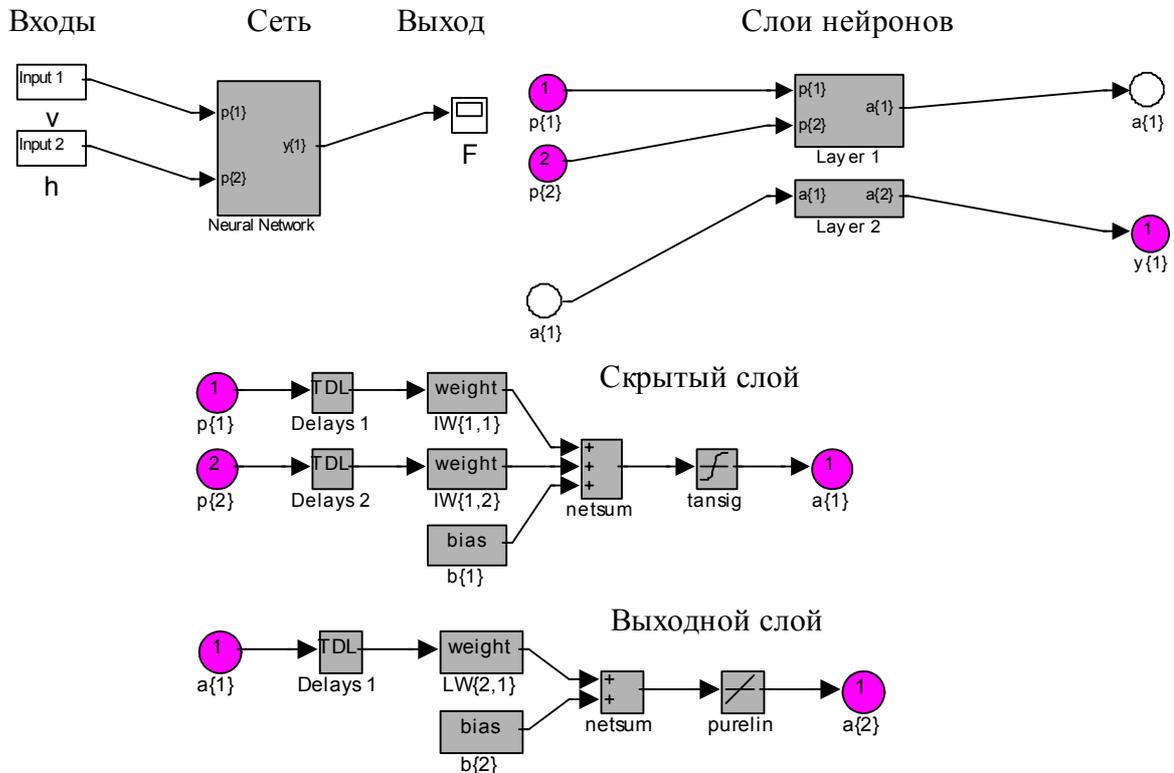


Рис. 10. Реализация нерекуррентной нейросетевой модели в MATLAB Neural Network Toolbox.

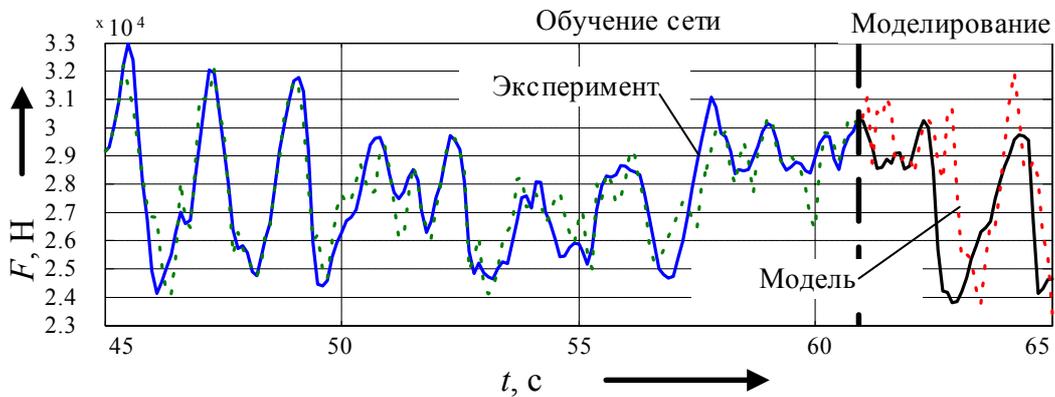


Рис. 11. Результаты обучения нерекуррентной сети и моделирования.

Вычислительный инструмент MATLAB Neural Network Toolbox не предназначен для моделирования и обучения сложных рекуррентных сетей. Поэтому использован дополнительный пакет расширения MATLAB Neural Network Based System Identification Toolbox (NNSYSID), разработанный в Техническом университете Дании, автор Magnus Nørgaard [6].

Выбрана нейросетевая модель авторегрессии — скользящего среднего с внешними входами (NNARMAX), имеющая обратные связи (рис. 12).

Количество задержанных выходов определено по корреляционной функции выходного сигнала $F(t)$ аналогично рис. 9. Выходной сигнал модели формируется с учетом его задержанных значений, а также задержанных значений ошибки $e(t)$.

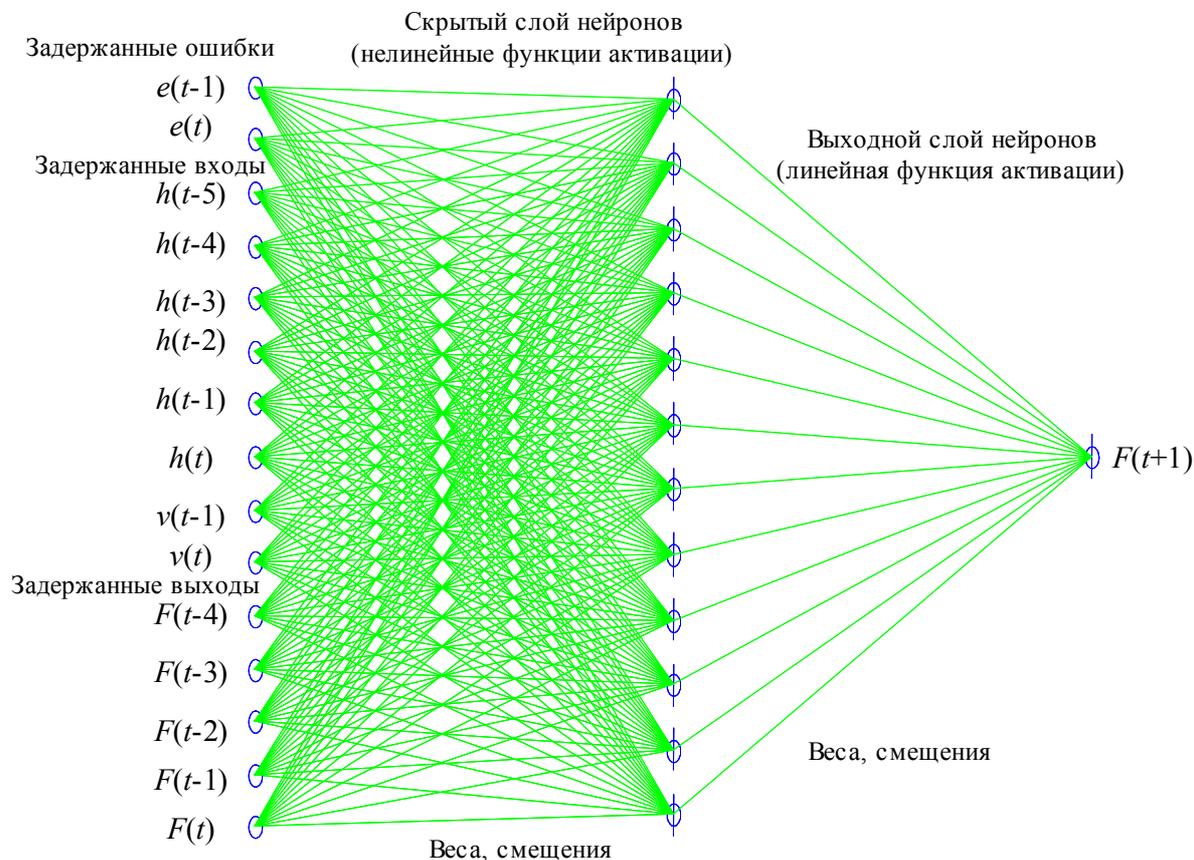


Рис. 12. Структура рекуррентной нейросетевой динамической модели, реализованной в пакете расширения MATLAB NNSYSID.

Предварительное масштабирование входных и выходных обучающих сигналов путем приведения их к нулевому математическому ожиданию и единичной дисперсии упрощает обучение нейросети (рис. 13). Введение обратных связей снижает время обучения на два порядка. Для обучения рекуррентной нейронной сети также применен метод Левенберга-Марквардта. Рекуррентная динамическая нейросетевая модель показывает лучшую точность прогнозирования (рис. 14).

Предлагаемая методика идентификации нелинейных динамических систем базируется на применении программной платформы MATLAB, обеспечивающей единый подход к формированию, представлению и цифровой обработке экспериментальных данных, открытость, гибкость при работе с большими массивами данных и одновременно набор мощных средств анализа и моделирования. Математическое моделирование на ос-

нове динамических нейронных сетей в MATLAB предоставляет новые возможности для исследования проблем управления и автоматизации.

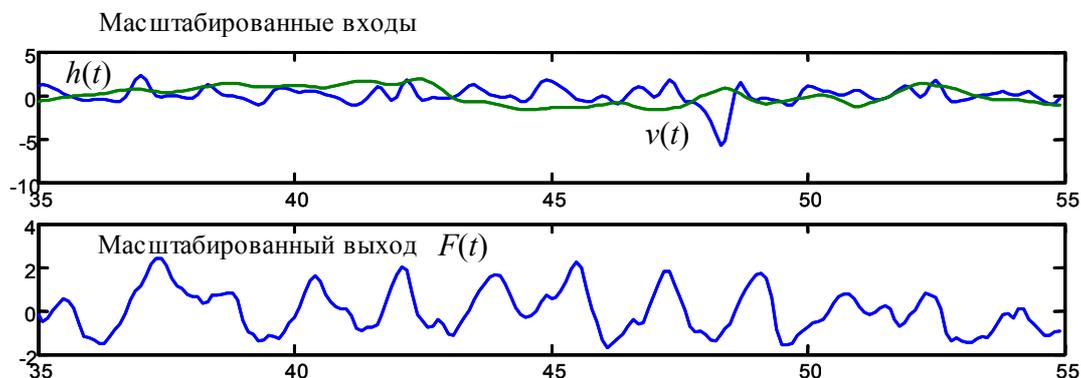


Рис. 13. Масштабирование входных и выходных обучающих сигналов в пакете расширения MATLAB NNSYSID.

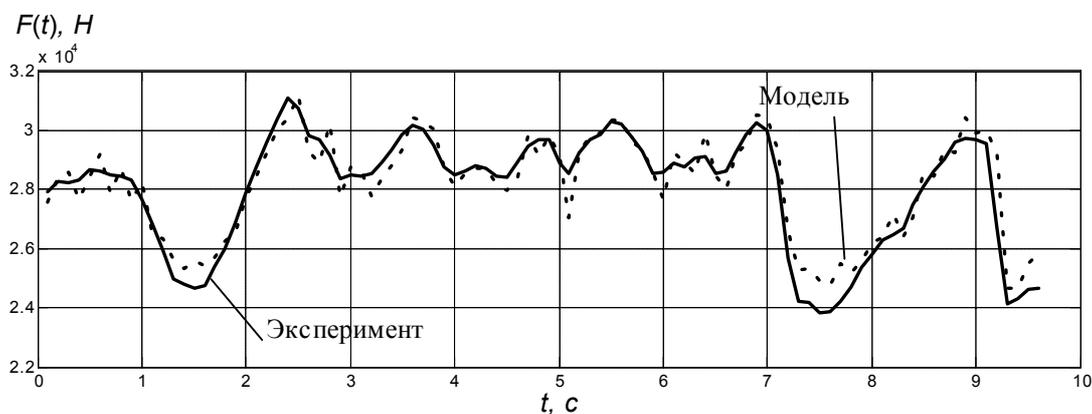


Рис. 14. Прогнозирование с помощью обученной рекуррентной нейросетевой динамической модели NNARMAX.

Литература

1. Интеллектуальные системы автоматического управления / Под ред. И. М. Макарова, В. М. Лохина.— М.: Физматлит, 2001.— 576 с.
2. Нейронные сети. STATISTICA Neural Networks.— М.: Горячая линия—Телеком, 2001.— 182 с.
3. Осовский С. Нейронные сети для обработки информации.— М.: Финансы и статистика, 2002.— 344 с.
4. Сергиенко А. Б. Цифровая обработка сигналов.— СПб.: Питер, 2002.— 608 с.
5. Медведев В. С., Потемкин В. Г. Нейронные сети. MATLAB 6.— М.: Диалог-МИФИ, 2002.— 496 с.
6. Nørgaard M. Neural Network Based System Identification Toolbox: Tech. Report. 97-E851, Department of Automation.— Technical University of Denmark, 1997.

УДК 621.391

СРАВНЕНИЕ ЭФФЕКТИВНОСТИ РЕАЛИЗАЦИИ АЛГОРИТМОВ ОБУЧЕНИЯ НЕЙРОСЕТИ В ЗАДАЧЕ ВОССТАНОВЛЕНИЯ ИЗОБРАЖЕНИЙ

Хрящёв В. В., Соколенко Е. А., Приоров А. Л.

*Ярославский государственный университет им. П. Г. Демидова, Ярославль,
e-mail: dcslab@uniyar.ac.ru*

1. Постановка задачи

Восстановление изображений и, в более общем смысле, восстановление сигналов относится к классу наиболее фундаментальных задач в современной науке. В общем случае модель искажения/восстановления изображения можно представить следующим образом [1] (рис. 1).

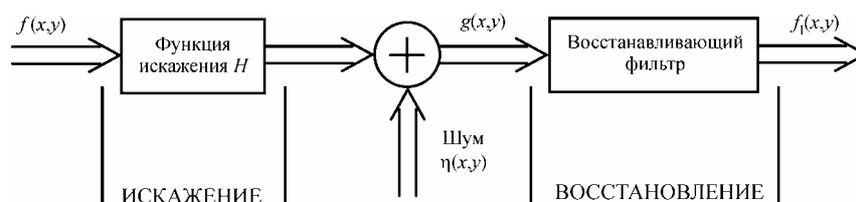


Рис. 1. Модель процесса искажения/восстановления изображений:

$f(x, y)$ — входное изображение; $g(x, y)$ — искаженное изображение; $f_1(x, y)$ — оценка входного изображения; H — функция искажения; $\eta(x, y)$ — аддитивный шум.

Задача восстановления состоит в получении оценочного изображения $f_1(x, y)$ по искаженному изображению $g(x, y)$ и имеющейся информации о H и $\eta(x, y)$. Если предположить, что H — линейный, инвариантный к сдвигу процесс, то в пространственной области искаженное изображение представляется следующим образом:

$$g(x, y) = h(x, y) * f(x, y) + \eta(x, y), \quad (1)$$

где $h(x, y)$ — пространственное представление искажающей функции; $*$ — обозначение свертки.

В частотной области, выражение (1) может быть записано следующим образом:

$$G(u, v) = H(u, v) \cdot F(u, v) + N(u, v).$$

Здесь заглавными буквами обозначены Фурье-трансформанты соответствующих функций выражения (1).

Так как, обычно мало что известно об искомом и искажающем сигналах, деконволюция двух сигналов в общем случае является сложной за-

дачей. Однако в частном случае, когда об искажающем сигнале известно, что фаза его Фурье-образа равна нулю, спектральная фаза искомого сигнала является неискаженной. Такая ситуация встречается, по крайней мере приблизительно, при длительном экспонировании через турбулентную атмосферу или в случае, когда изображения размыты из-за сильной дефокусировки линзы с круглыми диафрагмами.

Рассмотрим задачу восстановления сигнала по фазе его ДПФ как задачу аппроксимации, когда на входное воздействие (фаза ДПФ) требуется получить отклик (временной или пространственный сигнал) [8]. С этой точки зрения, учитывая, что нейронные сети очень эффективны при решении задач оценки, интерполяции, и позволяют аппроксимировать функции без использования их аналитического или близкого математического вида, можно ожидать, что с их помощью удастся решить задачу восстановления амплитуды [2].

Задача восстановления амплитуды относится к классу задач глобальной аппроксимации, и одной из лучших нейросетевых структур для решения подобных задач является многослойный персептрон (МСП) [3]. В данном случае использовался полносвязный МСП прямого распространения, имеющий входной, выходной и один скрытый слой. Для каждого нейрона в качестве нелинейной функции активации используется сигмоидная функция.

2. Методы обучения нейронной сети

Один из шагов построения нейросетевой модели представляет собой процесс, называемый обучением. Цель обучения состоит в выборе такого вектора весов w , чтобы минимизировать ошибку между фактическими реакциями нейрона y_i и ожидаемыми значениями z_i . Поэтому для обучения нейронной сети используется информация о текущем и ожидаемом значении выходного сигнала. Минимизация различий между фактическими реакциями нейрона y_i и ожидаемыми значениями z_i может быть представлена как минимизация конкретной функции погрешности (целевой функции) $E(w)$, чаще всего определяемой как

$$E(w) = \sum_{k=1}^p (y_i^{(k)}(w) - z_i^{(k)})^2,$$

где p — количество предъявляемых обучающих выборок.

Целевая функция $E(w)$ в общем случае является нелинейной функцией. По причине ее сложности часто используют итерационные алгоритмы для эффективного поиска решения в пространстве состояний. При этом алгоритм стартует от исходного значения w , которое потом корректируется. Последующая итерация w , обозначенная как w_n , определяется, как очередной шаг от текущей точки w_c в направлении вектора d ,

$$w_n = w_c + \eta d,$$

где положительное число η является величиной шага, показывающего, с какой скоростью мы движемся в направлении d . Каждый алгоритм обучения имеет свою схему корректировки весов нейронной сети. Рассмотрим некоторые из них.

2.1. Алгоритм обратного распространения

Одним из наиболее популярных алгоритмов обучения нейронных сетей является алгоритм обратного распространения (АОР), предложенный Румелхартом, Хинтоном и Виллиамсом (Rumelhart, Hinton, Williams) в 1986 г. АОР представляет собой алгоритм на основе пошагового принципа минимизации, когда веса нейронной сети корректируются в соответствии с направлением антиградиента в пространстве весов. Соответствующие формулы имеют вид:

$$\Delta w_c = w_n - w_c = -\eta \left. \frac{\partial E(w)}{\partial w} \right|_{w=w_c}, \quad (2)$$

где η является параметром скорости обучения (величиной шага). Формула (2) описывает групповое обучение, когда веса корректируются после предъявления на вход сети всех обучающих выборок.

Классический АОР, основанный на пошаговой минимизации, имеет такие существенные недостатки, как медленная сходимость и возможные осцилляции весов.

Важный путь повышения эффективности работы алгоритма обратного распространения заключается в использовании различных методов адаптации, делающих параметры обучения динамически настраиваемыми в процессе обучения.

2.2. Обучающие алгоритмы, использующие градиентные методы оптимизации

Поскольку процесс обучения нейронной сети можно рассматривать, как задачу оптимизации некоторой функции, то для ее решения можно использовать методы оптимизации более высокого порядка, использующие информацию о градиенте. Главная цель при этом — повышение скорости сходимости. В сравнении с эвристическими методами на основе алгоритма обратного распространения, эти методы хорошо теоретически обоснованы и обладают гарантированной сходимостью для большинства гладких функций.

Пусть, как и раньше d — вектор направления корректировки, η — коэффициент скорости обучения, w_c — текущее значение w . Тогда процесс обучения корректирует веса в соответствии с выражением

$$E(w_n) = E(w_c + \eta d) < E(w_c).$$

Принципиальное отличие между разными алгоритмами минимизации заключается в процедуре определения наилучшего направления d . Если направление выбрано, то оптимальная величина шага может быть выбрана при помощи поиска

$$\eta^* = \min_{\eta > 0} \varphi(\eta),$$

где $\varphi(\eta) = E(w_c + \eta d)$.

Если направление минимизации d определяется на основе градиента g целевой функции E , то такие методы оптимизации называются градиентными методами оптимизации.

2.2.1. Метод сопряженных градиентов

Метод сопряженных градиентов основан на квадратичной минимизации. При начальном значении градиента $g_i = \partial E / \partial w|_{w=w_i}$, векторе направления $d_i = -g_i$ метод сопряженных градиентов можно описать в виде двух рекурсивных выражений

$$\begin{aligned} g_n &= g_c + \lambda_c H d_c, \\ d_n &= -g_n + \gamma_c d_c, \end{aligned} \quad (3)$$

где

$$\lambda_c = \frac{g_c^T g_c}{d_c^T H d_c}, \quad (4)$$

$$\gamma_c = \frac{g_n^T g_n}{g_c^T g_c}, \quad (5)$$

или

$$\gamma_c = \frac{(g_n - g_c)^T g_n}{g_c^T g_c}. \quad (6)$$

При этом d называется сопряженным направлением, а H — матрицей Гессе целевой функции E . Выражение (5) называется формулой Флетчера-Ривса (Fletcher-Reeves), а выражение (6) — формулой Полака-Рибейры (Polak-Ribiere). Чтобы избежать необходимости знать матрицу Гессе для вычисления сопряженного направления осуществляют движение от w_c вдоль направления d_c до локального минимума E в точке w_n , а дальше полагают $g_n = \partial E / \partial w|_{w=w_n}$. Значение g_n можно использовать в качестве вектора (3) а выражение (4) больше не нужно. Таким образом, метод сопряженных градиентов весьма эффективен, особенно при обучении больших сетей.

При рассмотрении метода сопряженных градиентов применительно к обучению нейронных сетей можно сделать два важных вывода [7]. Довольно высока доля вычислений при одномерной оптимизации, поскольку каждое вычисление значения целевой функции приводит к вычислению отклика всей сети. Поэтому необходимо использовать эффективные алгоритмы одномерной оптимизации. Во-вторых, поскольку мы имеем дело с нейронными сетями, то функция ошибки не является квадратичной и, как следствие, сходимость зависит от степени соответствия функции ошибки ее квадратичной аппроксимации.

2.2.2. Квазиньютоновы алгоритмы обучения

Как и метод сопряженных градиентов, квазиньютоновы алгоритмы были получены в предположении квадратичной целевой функции. Для смещения направления градиента по методу Ньютона используется инвертированная матрица Гессе $B = H^{-1}$. При этом веса корректируются в соответствии с выражением:

$$w_n = w_c - \eta B_c g_c.$$

При этом матрица B вычисляется не «с нуля», а на основе матрицы на предыдущем шаге:

$$B_{now} = B_{old} + \Delta B_{now}.$$

Существуют две основных формулы для вычисления ΔB_{now} :

$$\Delta B_{now} = \frac{dd^T}{d^T \Delta g} - \frac{B_{old} \Delta g \Delta g^T B_{old}}{\Delta g^T B_{old} \Delta g}. \quad (7)$$

или

$$\Delta B_{now} = \left(1 + \frac{\Delta g^T B_{old} \Delta g}{d^T \Delta g} \right) \frac{dd^T}{d^T \Delta g} - \frac{d \Delta g^T B_{old} + B_{old} \Delta g d^T}{d^T \Delta g}. \quad (8)$$

где

$$d = w_{now} - w_{old}, \quad \Delta g = g_{now} - g_{old}.$$

Выражение (7) называется формулой Дэвидона-Флетчера-Пауэла (Davidon-Fletcher-Powell), а выражение (8) — формулой Бройдена-Флетчера-Голдфарба-Шанно (Broyden-Fletcher-Goldfarb-Shanno).

Стандартные квазиньютоновы методы требуют хранения N_w^2 переменных для выполнения аппроксимации обратной матрицы Гессе, кроме этого нужно выполнять одномерный поиск для вычисления требуемой длины шага. Здесь N_w — полное количество весовых коэффициентов нейронной сети. В сравнении с методом сопряженных градиентов квазиньютоновский алгоритм обладает более быстрой сходимостью, т. к. вместо вычисления гессиана выполняется его оценка.

2.2.3. Алгоритм Левенберга-Марквардта

Задача обучения нейронной сети обычно формулируется в виде нелинейной задачи наименьших квадратов. Соответственно для настройки параметров нейронной сети могут быть использованы методы, относящиеся к методам наименьших квадратов, например, метод Гаусса-Ньютона, который относится к методам линеаризации. Пусть r обозначает вектор, содержащий значения ошибки, J — якобиан, содержащий производные от r , имеющий N_w столбцов и $N_p \times N_y$ строк, где N_p — число выборок, N_y — число выходов. Выражение, описывающее метод Гаусса-Ньютона, имеет вид:

$$w_n = w_c - (J_c^T J_c)^{-1} (J_c^T r_c).$$

Если J_c является неполной, то можно применять метод Левенберга-Марквардта (Levenberg-Marquardt), при этом выражения для корректировки весов примет вид:

$$w_n = w_c - (J_c^T J_c + \mu I)^{-1} (J_c^T r_c).$$

где μ — неотрицательное число. В модифицированном алгоритме Левенберга-Марквардта, который был предложен для обучения многослойных сетей прямого распространения используется диагональную матрицу вместо единичной матрицы I . Вычислительную сложность и требования к памяти у алгоритма Левенберга-Марквардта можно уменьшить, если использовать неполные матрицы Якоби.

2.2.4. Алгоритмы глобальной оптимизации

Другой важный класс методов использует методы случайной оптимизации, которые характеризуются наличием элемента случайного поиска в процессе обучения, что позволяет не останавливаться в локальных минимумах, а сходиться к глобальному минимуму целевой функции.

К методам глобальной оптимизации относятся генетические алгоритмы обучения нейросети [6]. Эти алгоритмы имитируют процессы наследования свойств живыми организмами и генерируют последовательности новых векторов w , содержащие оптимизированные значения параметров. При этом выполняются операции трех видов: селекция, скрещивание и мутация. Идея этих операций, как и их название, заимствованы из генетики: отбор наиболее приспособленных хромосом — векторов w (селекция), обмен комплементарными частями среди отобранных хромосом (скрещивание) и замена значений отдельных генов случайными допустимыми значениями (мутация).

Исследованиями доказано [6], что каждое последующее поколение, сформированное после выполнения указанных операций, имеет статисти-

чески лучшие средние показатели приспособленности. Когда обучение считается завершенным, в качестве окончательного решения принимается наиболее приспособленная хромосома. Решение об остановке процесса обучения может быть принято либо при достижении удовлетворительного уровня ошибки, либо при выполнении максимального количества итераций, либо в случае отсутствия прогресса минимизации.

Поскольку методы случайного поиска, в том числе и генетические, сами по себе сходятся достаточно медленно, то целесообразно рассматривать более общие методы, являющиеся объединением традиционных градиентных методов обучения и методов случайной оптимизации. Если в процессе обучения градиентным методом встречается пологий участок целевой функции, то обучающий алгоритм переключается на метод случайной оптимизации. После того, как пологий участок пройден, алгоритм вновь переключается на градиентный метод.

3. Реализация рассмотренных методов в среде MATLAB

Поскольку для решения задачи используется двухслойная сеть прямого распространения, то для создания сети используется функция `net = newff(PR,[S1 S2...SNI],{TF1 TF2...TFNI},btf,bf,pf)`

где `PR` — массив размера $R \times 2$ минимальных и максимальных значений для R векторов входа; `Si` — количество нейронов в слое i ; `Tfi` — функция активации слоя i , по умолчанию `tansig`; `btf` — обучающая функция, реализующая метод обратного распространения, по умолчанию `trainlm`; `bf` — функция настройки, реализующая метод обратного распространения, по умолчанию `learngdm`; `pf` — критерий качества обучения, по умолчанию `mse`; `net` — объект класса `network object` — нейронная сеть.

В рассматриваемом случае вызов функции `newff()` выглядит следующим образом:

```
min_max = zeros(Ni,2);
min_max(:,1) = -pi;
min_max(:,2) = pi;
net = newff(min_max,[hls No],{'logsig','logsig'],'traingd');
```

Здесь `Ni` — количество входов нейронной сети, `hls` — количество нейронов в скрытом слое, `No` — количество выходов нейронной сети. В приведенной записи предполагается, что для обучения будет использоваться классический алгоритм обратного распространения. Этот алгоритм реализует функция пакета `Neural Network Toolbox` ‘`traingd`’.

Функция `traingd` характеризуется следующими параметрами, заданными по умолчанию:

```
net.trainparam
ans =
    epochs:    100
    goal:      0
```

```
lr: 1.0000e-002
max_fail: 5
min_grad: 1.0000e-010
show: 25
time: Inf
```

Здесь `epochs` — максимальное количество циклов (эпох) обучения; `goal` — предельное значение критерия обучения; `lr` — параметр скорости настройки; `max_fail` — максимально допустимый уровень превышения ошибки контрольного подмножества по сравнению с обучающим; `min_grad` — минимальное значения градиента; `show` — интервал вывода информации, измеренный в циклах; `time` — предельное время обучения в секундах.

Результаты тестирования данного метода при решении рассматриваемой задачи показали, что алгоритм не достиг заданного уровня ошибки даже через 1000 эпох обучения. Алгоритмы второго порядка сходятся гораздо быстрее.

Рассмотрим методы сопряженных градиентов. Функция `'traincgf'` реализует алгоритм Флетчера-Ривса, а функция `'traincgp'` — алгоритм Полака-Рибейры. Эти функции характеризуются следующими параметрами по умолчанию:

```
net = newff(min_max,[hls No],{'logsig','logsig'},'traincgp');
net.trainParam
ans =
```

```
epochs: 100
show: 25
goal: 0
time: Inf
min_grad: 1.0000e-006
max_fail: 5
searchFcn: 'srchcha'
scale_tol: 20
alpha: 0.0010
beta: 0.1000
delta: 0.0100
gamma: 0.1000
low_lim: 0.1000
up_lim: 0.5000
maxstep: 100
minstep: 1.0000e-006
bmax: 26
```

Здесь `SearchFcn` — имя функции одномерного поиска; `scale_tol` — коэффициент для вычисления шага процедуры одномерного поиска; `alpha` — коэффициент, определяющий порог уменьшения критерия качества; `beta` — коэффициент, определяющий выбор шага; `delta` — начальный шаг разбиения интервала; `gamma` — параметр, регулирующий изменение критерия качества; `lo_lim` — нижняя граница изменения шага; `up_lim` — верхняя граница изменения шага; `maxstep` — максимальное значение шага;

`minstep` — минимальное значение шага; `bmax` — максимальное значение шага для процедуры `srchhyb`.

Алгоритм Бройдена-Флетчера-Голдфарба-Шанно реализован в MATLAB в виде функции обучения '`traindfg`'. Параметры данной функции практически совпадают с параметрами функции `traincgf`, за исключением программы одномерного поиска, которая в данном случае заменена М-функцией `srchbas`.

Последний алгоритм, который тестировался при решении рассматриваемой задачи — это алгоритм Левенберга-Марквардта (функция обучения `trainlm`) и его модификация на основе метода регуляризации Байеса (`trainbr`). Параметры по умолчанию этих функций приведены ниже:

`net.trainParam`

`ans =`

```

epochs:    100
goal:      0
max_fail:  5
mem_reduc: 1
min_grad:  1.0000e-010
mu:        0.0010
mu_dec:    0.1000
mu_inc:    10
mu_max:    1.0000e+010
show:      25
time:      Inf

```

Рассмотрим новые параметры. Параметр `mu` — начальное значение для коэффициента μ . Это значение умножается либо на коэффициент `mu_dec`, когда функционал ошибки уменьшается, либо на коэффициент `mu_inc`, когда функционал ошибки возрастает. Если `mu` превысит значение `mu_max`, алгоритм останавливается. Параметр `mem_reduc` позволяет экономить объем используемой памяти (чем его значение больше, тем меньше объем требуемой памяти и тем меньше быстродействие алгоритма).

В табл. 1 представлены результаты, полученные на компьютере Intel Celeron 900МГц, позволяющие сравнить длительность, количество циклов обучения и вычислительную сложность различных алгоритмов (во всех случаях обучение проводилось до получения на выходе нейронной сети среднеквадратичной ошибки (`sse`) меньшей, чем 0.5). Все алгоритмы были реализованы в среде MATLAB 6.5 (Release 13), что создало основу для получения объективных оценок применительно к данной конкретной задаче. В рассматриваемом случае нейронная сеть имела три слоя. Количество нейронов во входном и скрытом слоях было равно 16, в выходном слое — 8. Это соответствует разбиению изображения на блоки по 8 пикселей и вычислению для каждого из них 32-точечного ДПФ [8].

Таблица 1.
Сравнение эффективности алгоритмов обучения.

Название алгоритма	Время обучения, с	Количество циклов обучения	Число операций	Максимальная энергия ошибки для 10000 тестовых сигналов
Алгоритм Левенберга–Марквардта	20	20	8 135	1.66
Алгоритм, основанный на методе Байеса	60	43	18 630	0.8
Алгоритм на основе метода масштабируемых сопряженных градиентов	130	1492	611 998	2.6
Алгоритм на основе метода Флетчера–Ривса	300	2708	1 483 726	1.8
Алгоритм на основе метода Полака–Рибейры	88	814	447 646	1.7
Алгоритм на основе метода Пауэлла–Биеле	211	1839	1 102 299	2.1

Выводы

По результатам многочисленных и различных тестов были сделаны следующие выводы:

Алгоритмы, основанные на алгоритме обратного распространения при решении данной задачи дают неудовлетворительные результаты, ни один из них не позволяет получить приемлемое значение ошибки на выходе нейронной сети за удовлетворительное время обучения.

Наибольшую эффективность показал алгоритм Левенберга–Марквардта. При его использовании наблюдалось наименьшее время обучения, наименьшее количество циклов обучения и наименьшая вычислительная сложность.

Следующим по эффективности обучения идет метод Полака–Рибейры, относящийся к группе методов сопряженных градиентов. Этому методу немного уступает метод масштабируемых сопряженных градиентов.

Отдельно следует сказать про алгоритм, основанный на методе Байеса. Фактически эта процедура обучения реализует алгоритм Левенберга–Марквардта, дополненный методом регуляризации, уменьшающим эффект переобучения. Из приведенной таблицы видно, что процедура настройки параметров сети длится втрое дольше, при этом выполняется приблизительно вдвое больше циклов обучения, чем для «чистого» метода Левенберга–Марквардта, но способность сети к обобщению (эффективность аппроксимации реальной зависимости) оказывается наилучшей. В рассматриваемой задаче целесообразно использовать именно эту процедуру обу-

чения, поскольку, во-первых, метод Левенберга-Марквардта оказался наиболее эффективным при настройке параметров сети на обучающем множестве и, во-вторых, не предъявляется жестких требований к скорости обучения сети.

Был также проведен ряд экспериментов по обучению нейросети генетическим методом обучения. При этом использовался пакет расширения Genetic Algorithm Toolbox [9], а также разработанные авторами алгоритмы. Согласно полученным результатам, генетические алгоритмы целесообразно использовать, когда используемый для обучения итерационный алгоритм застревает в локальном минимуме. Использование генетического алгоритма самого по себе не позволяет получить требуемую ошибку на выходе сети за удовлетворительное время.

Литература

1. *Gonzalez R., Woods R.* Digital Image Processing.— Prentice-Hall, 2002.
2. *Осовский С.* Нейронные сети для обработки информации / Пер. с польского И. Д. Рудинского.— М.: Финансы и статистика, 2002.— 344 с.
3. *Ланнэ А. А.* Нейронные цепи, тринадцатая проблема Гильберта и задачи обработки сигналов // Вестник молодых ученых.— 2001.— №7.— С.3-26.
4. *Медведев В. Г., Потемкин В. Г.* Нейронные сети. MATLAB 6.— М.: Диалог–МИФИ, 2002.— 496 с.
5. *Банди Б.* Методы оптимизации. Вводный курс: Пер. с англ.— М.: Радио и связь, 1988.— 128 с.
6. *Goldberg D. E.* Genetic Algorithms in Search, Optimization and Machine Learning.— Addison Wesley Publishing Company, January 1989.
7. *Karayiannis N. B., Venetsanopoulos A. N.* Artificial neural networks: learning algorithms, performance evaluation, and applications.— Kluwer Academic, Boston, MA, 1993.
8. *Хрящев В. В., Соколенко Е. А., Приоров А. Л.* Нейросетевое восстановление амплитуды дискретного сигнала по его фазовому спектру // Доклады 5-й Международной конф. «Цифровая обработка сигналов и ее применение».— М., 2003.— С.622–624.
9. <http://www.shef.ac.uk/~gaipp/ga-toolbox/>.

УДК 658.012

ПРОЕКТИРОВАНИЕ НЕЧЕТКИХ КЛАССИФИКАТОРОВ В СИСТЕМЕ MATLAB

Штовба С. Д., Панкевич О. Д.

Винницкий национальный технический университет, Винница, Украина,

e-mail: shtovba@ksu.vstu.vinnica.ua

Введение

Классификация на основе нечеткого логического вывода используется при принятии решений в технике, экономике, политике, медицине, биологии и в других областях [1-3]. Системы нечеткого вывода базируются на лингвистических правилах «Если —То». Они обеспечивают хороший баланс между безошибочностью классификации и прозрачностью модели принятия решения.

Пакет **Fuzzy Logic Toolbox** вычислительной системы MATLAB предоставляет широкий набор инструментов для проектирования и исследования систем нечеткого логического вывода с непрерывным выходом. В настоящей статье показано, как расширить **Fuzzy Logic Toolbox** для проектирования нечетких классификаторов, т. е. систем нечеткого логического вывода с дискретным выходом.

Статья организована следующим образом: в разделе 1 излагаются математические модели нечеткого вывода для задач классификации; в разделе 2 ставятся задачи настройки нечеткого классификатора по различным критериям обучения; в разделе 3 приводится программа нечеткой классификации, использующая функции пакета **Fuzzy Logic Toolbox**; в разделе 4 предлагаются программы настройки нечеткого классификатора, использующие **Optimization Toolbox**; в разделе 5 приводятся примеры проектирования нечетких классификаторов с применением предложенных программных средств.

1. Нечеткий классификатор

Будем рассматривать классификатор с n входами (x_1, x_2, \dots, x_n) и одним выходом y (рис. 1), что соответствует отображению вида:

$$X = (x_1, x_2, \dots, x_n) \rightarrow y \in \{d_1, d_2, \dots, d_m\},$$

где d_1, d_2, \dots, d_m - классы (типы решений).

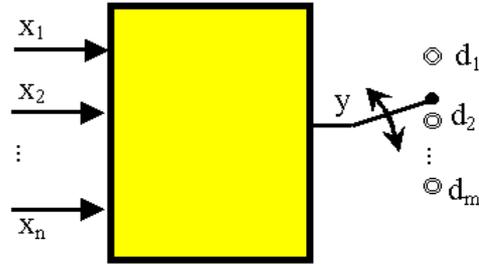


Рис. 1. Классификатор.

Классификация на основе нечеткого логического вывода осуществляется по следующей базе знаний [2, 3]:

ЕСЛИ $(x_1 = a_{1,j1})$ И $(x_2 = a_{2,j1})$ И ... И $(x_n = a_{n,j1})$ с весом w_{j1} ,
ИЛИ $(x_1 = a_{1,j2})$ И $(x_2 = a_{2,j2})$ И ... И $(x_n = a_{n,j2})$ с весом w_{j2} ,
 ...
ИЛИ $(x_1 = a_{1,jk_j})$ И $(x_2 = a_{2,jk_j})$ И ... И $(x_n = a_{n,jk_j})$ с весом w_{jk_j} ,
ТО $y = d_j$, $j = \overline{1, m}$, (1)

где $a_{i,jp}$ — нечеткий терм, которым оценивается переменная x_i в строчке с номером jp ($p = \overline{1, k_j}$), т. е. $a_{i,jp} = \int_{[x_i, \bar{x}_i]} \mu_{jp}(x_i)/x_i$; k_j — количество строчек-конъюнкций, в которых выход y оценивается значением d_j ; $w_{jp} \in [0, 1]$ — весовой коэффициент правила с номером jp .

Степени принадлежности объекта $X^* = (x_1^*, x_2^*, \dots, x_n^*)$ классам d_j рассчитываются так [2, 3]:

$$\mu_{d_j}(X^*) = \bigvee_{p=1, k_j} w_{jp} \cdot \bigwedge_{i=1, n} (\mu_{jp}(x_i^*)), \quad j = \overline{1, m}, \quad (2)$$

где $\mu_{jp}(x_i^*)$ — степень принадлежности входа x_i^* нечеткому терму $a_{i,jp}$; \bigvee (\bigwedge) — s-норма (t-норма), которой в задачах классификации обычно соответствует максимум (минимум).

В качестве решения выбирают класс с максимальной степенью принадлежности:

$$y^* = \arg \max_{\{d_1, d_2, \dots, d_m\}} (m_{d_1}(X^*), m_{d_2}(X^*), \dots, m_{d_m}(X^*)). \quad (3)$$

Пример 1.1. Известна нечеткая база знаний:

Если $x_1 = \text{низкий}$	И	$x_2 = \text{низкий}$,	то $y = \text{класс 1}$;
Если $x_1 = \text{средний}$	И	$x_2 = \text{высокий}$,	то $y = \text{класс 2}$;
Если $x_1 = \text{высокий}$	И	$x_2 = \text{высокий}$,	то $y = \text{класс 3}$;
Если $x_1 = \text{высокий}$	И	$x_2 = \text{низкий}$,	то $y = \text{класс 2}$.

На рис. 2 приведены результаты классификации 600 объектов при реализации t-нормы операцией минимума и s-нормы операцией максимума. Области, соответствующие первому, второму, третьему и четвертому правилам базы знаний обозначены на рисунке символами #1, #2, #3 и #4.

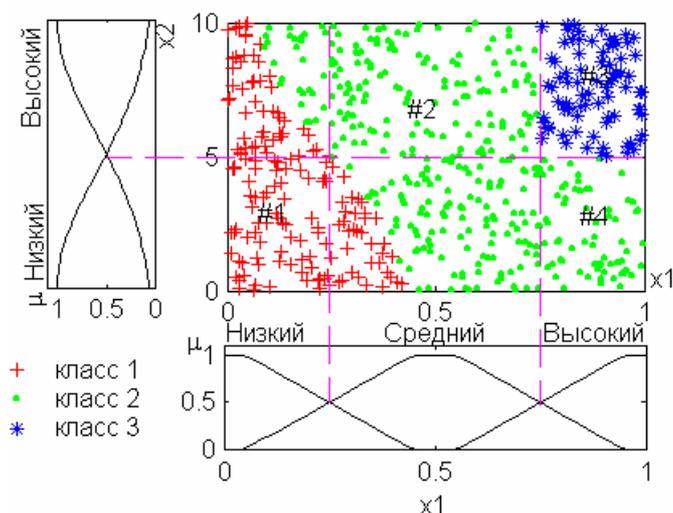


Рис. 2. Классификации по нечеткой базе знаний из примера 1.1

2. Постановки задач настройки нечеткого классификатора

Будем предполагать, что существует обучающая выборка из M пар экспериментальных данных, связывающих входы $X = (x_1, x_2, \dots, x_n)$ с выходом y исследуемой зависимости:

$$(X_r, y_r), \quad (r = \overline{1, M}), \quad (4)$$

где $X_r = (x_{r,1}, x_{r,2}, \dots, x_{r,n})$ — входной вектор в r -ой паре и y_r — соответствующий выход.

Настройка представляет собой нахождение таких параметров функций принадлежности термов входных переменных и весовых коэффициентов правил, которые минимизируют отклонение между желаемым и действительным поведением нечеткого классификатора на обучающей выборке. Критерий близости можно определить различными способами.

Первый способ заключается в выборе в качестве критерия настройки процента ошибок классификации на обучающей выборке. Введем следующие обозначения:

P — вектор параметров функций принадлежности термов входных и выходной переменных;

W — вектор весовых коэффициентов правил базы знаний;

$F(X_r, P, W)$ — результат вывода по нечеткой базе (1) с параметрами (P, W) при значении входов X_r . Нечеткий логический вывод осуществляется по формулам (2)–(3).

Тогда настройка нечеткого классификатора сводится к следующей задаче оптимизации: *найти такой вектор (P, W) , чтобы:*

$$\frac{1}{M} \sum_{r=1, M} \Delta_r \rightarrow \min, \quad (5)$$

где Δ_r — ошибка классификации объекта X_r :

$$\Delta_r = \begin{cases} 1, & \text{если } y_r \neq F(X_r, P, W) \\ 0, & \text{если } y_r = F(X_r, P, W) \end{cases}.$$

Преимущества критерия настройки (5) заключаются в его простоте и ясной содержательной интерпретации. Процент ошибок широко используется как критерий обучения различных систем распознавания образом. Целевая функция задачи оптимизации (5) принимает дискретные значения. Это затрудняет использование градиентных методов оптимизации, т. к. на протяженных плато целевой функции алгоритмы оптимизации «застревают». Особенно трудно подобрать подходящие параметры градиентных алгоритмов (например, приращения аргументов для расчета частных производных) при настройке нечеткого классификатора на небольшой выборке данных.

Второй способ использует в качестве критерия настройки расстояние между результатом вывода в виде нечеткого множества $\left(\frac{\mu_{d_1}(X)}{d_1}, \frac{\mu_{d_2}(X)}{d_2}, \dots, \frac{\mu_{d_m}(X)}{d_m} \right)$ и значением выходной переменной в обучающей выборке. Для этого выходную переменную y в обучающей выборке (4) фаззифицируют следующим образом [2, 3]:

$$\left. \begin{aligned} \tilde{y} &= (1/d_1, 0/d_2, \dots, 0/d_m), & \text{если } y = d_1 \\ \tilde{y} &= (0/d_1, 1/d_2, \dots, 0/d_m), & \text{если } y = d_2 \\ &\dots \\ \tilde{y} &= (0/d_1, 0/d_2, \dots, 1/d_m), & \text{если } y = d_m \end{aligned} \right\} \quad (6)$$

В этом случае настройка нечеткого классификатора сводится к следующей задаче оптимизации [2, 3]: *найти такой вектор (P, W) , чтобы:*

$$\frac{1}{M} \cdot \sum_{r=1}^M \sum_{j=1}^m \left(m_{d_j}(y_r) - m_{d_j}(X_r, P, W) \right)^2 \rightarrow \min, \quad (7)$$

где $m_{d_j}(y^r)$ — степень принадлежности значения выходной переменной y в r -ой пары обучающей выборке к решению d_j в соответствии с (6); $m_{d_j}(X_r, P, W)$ — степень принадлежности выхода нечеткой модели с параметрами (P, W) к решению d_j , определяемая по формуле (2) при значениях входов из r -ой пары обучающей выборке (X_r) .

Целевая функция в задаче (7) не имеет протяженных плато, поэтому

она пригодна к оптимизации градиентными методами. Однако, результаты оптимизации не всегда удовлетворительные: нечеткая база знаний, обеспечивающая минимум критерия (7), не всегда обеспечивает также и минимум ошибок классификации. Это объясняется тем, что точки, близкие к границам раздела классов, вносят почти одинаковый вклад в критерий настройки, как при правильной, так и при ошибочной классификации.

Третий способ наследует достоинства предыдущих способов. Идея заключается в том, чтобы вклад ошибочно классифицированных объектов в критерий настройки увеличивать, посредством умножением расстояния

$\sum_{j=1}^m (m_{d_j}(y_r) - m_{d_j}(X_r, P, W))^2$ на штрафной коэффициент. В результате задача оптимизации принимает следующий вид:

$$\frac{1}{M} \cdot \sum_{r=1}^M (\Delta_r \cdot \text{penalty} + 1) \cdot \sum_{j=1}^m (m_{d_j}(y_r) - m_{d_j}(X_r, P, W))^2 \rightarrow \min, \quad (8)$$

где $\text{penalty} > 0$ — штрафной коэффициент.

Задачи (5), (7) и (8) могут быть решены различными технологиями оптимизации, среди которых часто применяется метод наискорейшего спуска, квазиньютоновские методы и генетические алгоритмы. На управляемые переменные P обычно накладывают ограничения, обеспечивающие линейную упорядоченность элементов терм-множеств. Такие ограничения не позволяют алгоритмам оптимизации сделать, например, нечеткое множество «Низкий» больше «Высокого». Кроме того, ядра нечетких множеств не должны выходить за пределы диапазонов изменения соответствующих переменных. Это обеспечивает прозрачность, т. е. содержательную интерпретабельность нечеткой базы знаний после настройки. Что касается вектора W , то его координаты должны находиться в диапазоне $[0, 1]$. Если к уровню интерпретабельности базы знаний предъявляются высокие требования, то веса правил не настраивают, оставляя их равными 1. Возможен и промежуточный вариант, когда весовые коэффициенты могут принимать значения 0 и 1. В этом случае нулевое значения весового коэффициента эквивалентно исключению правила из нечеткой базы знаний.

Параметры функций принадлежности и веса правил можно настраивать одновременно или по отдельности. При настройке только весов правил объем вычислений можно значительно сократить, т. к. степени принадлежности $\mu_{jp}(x_i^*)$, входящие в (2), не зависят от W . Для этого в начале оптимизации надо рассчитать степени выполнения правил при единичных весовых коэффициентах ($w_{jp} = 1$) для каждого объекта из обучающей выборки:

$$g_{jp}(X_r) = \bigwedge_{i=1, n} \mu_{jp}(x_{r,i}), \quad j = \overline{1, m}, \quad p = \overline{1, k_j}, \quad r = \overline{1, M}.$$

Для новых весовых коэффициентов степени принадлежности объекта X_r классам d_j рассчитываются так:

$$\mu_{d_j}(X_r) = \bigvee_{p=1, k_j} w_{jp} \cdot g_{jp}(X_r), \quad j = \overline{1, m}. \quad (9)$$

3. Классификация на основе нечеткого вывода в системе MATLAB

Пакет **Fuzzy Logic Toolbox** [4] вычислительной системы MATLAB обеспечивает проектирование систем нечеткого логического вывода для объектов с непрерывным выходом, т. е. для случая, когда выходная переменная y может принимать значения из диапазона $[\underline{y}, \overline{y}]$. В настоящей разделе показывается, как расширить **Fuzzy Logic Toolbox** для выполнения классификации на основе нечеткого вывода.

В качестве нечеткого классификатора предлагается использовать систему нечеткого логического вывода типа Сугено. Классам решений $\{d_1, d_2, \dots, d_m\}$ ставятся в соответствие термы выходной переменной; наименование класса решений зададим как элемент терм-множества выходной переменной. Параметры заключений правил (параметры «функций принадлежности» выходной переменной) могут быть произвольными, т. к. они не влияют на результат классификации. Проектирование системы нечеткого вывода типа Сугено можно удобно осуществлять в редакторе **fuzzy** из **Fuzzy Logic Toolbox**.

Для выполнения нечеткой классификации нами разработана функция **fuzzy_classifier**, листинг которой приведен в Приложении 1. Функция нечеткой классификации вызывается в таком формате:

decision=fuzzy_classifier(X, fis, type),

где **X** — вектор информативных признаков объекта классификации; **fis** — система нечеткого логического вывода; **type** — тип возвращаемого функцией результата (допустимые значения: **'number'** — порядковый номер класса и **'name'** — имя класса. Значение по умолчанию — **'number'**); **decision** — результат классификации для объекта **X**.

Функция **fuzzy_classifier** вызывает функцию **evalfis** в формате

[a, b, c, d] = evalfis(x, fis),

что позволяет получить промежуточные результаты нечеткого логического вывода. Затем находятся правила с максимальной степенью выполнения. Если таких правил несколько, тогда среди конкурирующих классов выбирает тот, сумма степеней принадлежности которому максимальна.

Функция нечеткой классификации может вызываться с двумя выходными аргументами:

[decision, mf_grades]=fuzzy_classifier(X, fis, type),

где **mf_grades** — вектор степеней принадлежности объекта **X** классам решений.

4. Настройка нечеткого классификатора в системе MATLAB

Настройка нечеткого классификатора сводится к задачам оптимизации (5), (7) или (8). Для решения этих задач в системе MATLAB можно воспользоваться пакетом **Optimization Toolbox**. На основе программ обучения нечетких моделей типа Мамдани [5] нами разработаны типовые сценарии настройки нечетких классификаторов, а также типовые целевые функции. М-файлы этих программ приведены на сайте www.MATLAB.ru в разделе Fuzzy Logic Toolbox.

Для быстрой настройки весов правил нечеткого классификатора на основе соотношения (9) авторами написан пакет FALEFC (**FA**st **LE**arning the **F**uzzy **C**lassifier). Время настройки весов правил с помощью пакета FALEFC на порядок меньше, чем при настройке функций принадлежности. Пакет FALEFC использует функции **Fuzzy Logic Toolbox** и **Optimization Toolbox**. Пакет включает такие программы:

- fuzzy_classifier** - выполнение нечеткого вывода для задач классификации;
- fast_w_learning** - сценарий быстрой настройки весов правил нечеткого классификатора;
- dp_for_fuzzy_cl_learning** - считывание выборки данных и преобразование ее к формату, требуемому программами настройки нечеткого классификатора;
- rmg_fuzzy_cl** - расчет RMG - матрицы степеней выполнения правил при единичных весовых коэффициентах, необходимой для быстрой классификации;
- fuzzy_classifier_rmg** - быстрая классификация на основе нечеткого вывода при новых значениях весов правил с использованием RMG (матрицы степеней выполнения правил при единичных весовых коэффициентах);
- ob_fun_fast_w** - целевая функция для быстрой настройки весов нечетких правил, использующая RMG;
- fast_cl_testing_with_rmg** - быстрое тестирование нечеткого классификатора при новых весах правил с использованием RMG;
- rule_order_fuzzy_cl** - определение количества правил в базе знаний для каждого класса (типа решения).

Пакет FALEFC также доступен в разделе Fuzzy Logic Toolbox сайта www.MATLAB.ru.

5. Примеры проектирования нечетких классификаторов в системе MATLAB

Пример 5.1. Рассматривается задача классификации ирисов, предложенная Фишером в 1936 г. Задача состоит в отнесении ириса к одному из трех классов: 1 - *Iris Setosa*, 2 - *Iris Versicolor*; 3 - *Iris Virginica*. При классификации используются следующие признаки цветков: x_1 - длина чашелистика; x_2 - ширина чашелистика; x_3 - длина лепестка; x_4 - ширина лепестка. Исходные данные для классификации ирисов записаны в файле `iris.dat`, входящего в **Fuzzy Logic Toolbox**. Файл содержит 150 строк, каждая из которых описывает один ирис. Информация о цветке представлена пятеркой чисел - первые четыре числа соответствуют значениям признаков, а пятое — классу ириса. Двумерные распределения ирисов показаны на рис. 3.

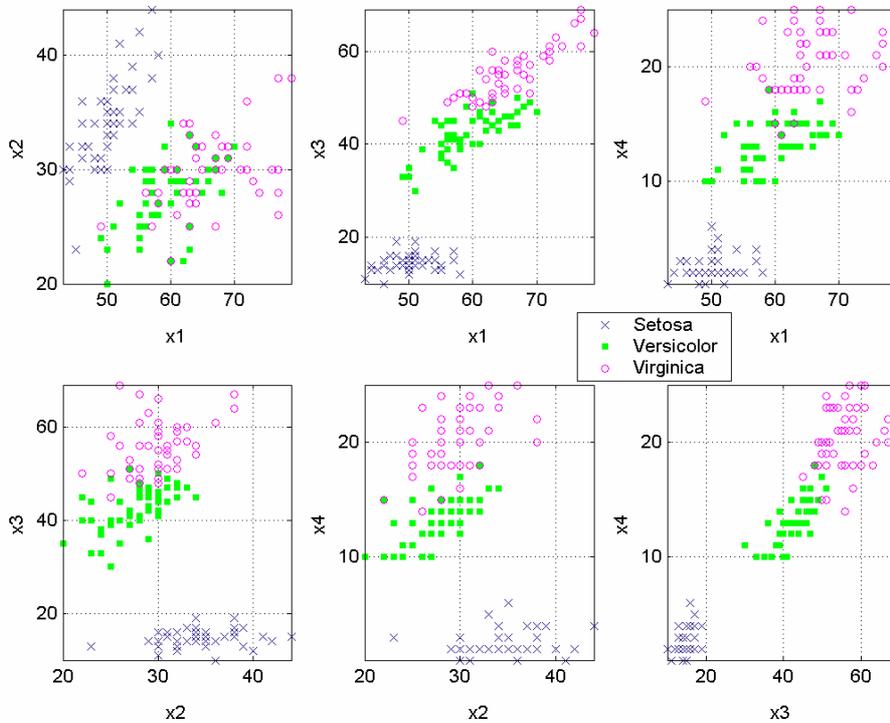


Рис. 3. 2D-распределения фишеровских ирисов.

В редакторе **fuzzy** создадим систему нечеткого логического вывода типа Сугено с четырьмя входными и одной выходной переменными. Диапазоны изменения входных переменных установим такими же, как и для исходных данных: $x_1 \in [47, 79]$; $x_2 \in [20, 44]$; $x_3 \in [10, 69]$; $x_4 \in [1, 25]$. Для лингвистической оценки признаков цветков будем использовать термы «низкий», «средний» и «высокий» с установленными по умолчанию треугольными функциями принадлежности. Взаимосвязь «входы - выход» опишем тремя нечеткими правилами:

если x_4 = 'низкий',

то y = '*Iris Setosa*' ;

если $x_3 = \text{'средний'}$ и $x_4 = \text{'средний'}$, то $y = \text{'Iris Versicolor'}$;
 если $x_3 = \text{'высокий'}$ и $x_4 = \text{'высокий'}$, то $y = \text{'Iris Virginica'}$.

Логической операций «И» в посылках правил поставим в соответствие операцию минимума над функциями принадлежности. Для этого в редакторе **fuzzy** выберем опцию **min** в меню **And method**.

Применяя функцию **fuzzy_classifier** обнаруживаем, что созданная нечеткая модель правильно классифицирует 130 из 150 ирисов. Для настройки весов правил сформируем обучающую и тестирующую выборки. В обучающую выборку включим 120 ирисов с порядковыми номерами не кратными 5. В связи с небольшим количеством данных тестировать нечеткий классификатор будем на всей выборке. Для быстрой настройки весов правил нечеткого классификатора в типовом сценарии **fast_w_learning** укажем имена файлов системы нечеткого вывода **iris_cl.fis** и экспериментальных данных **iris.dat**. Сценарий настройки нечеткого классификатора ирисов приведен в Приложении 2.

В результате настройки получены такие весовые коэффициенты: 0.9 — для первого правила; 0.1 — для второго правила; 0.8 — для третьего правила. Настроенная система нечеткого логического вывода правильно классифицирует 116 ирисов (96.67%) из обучающей выборки и 144 (96%) ирисов из всей выборки цветков.

Пример 5.2. Рассматривается объект с двумя входами $x_1, x_2 \in [0, 10]$ и одним выходом y , который может принимать одно из трех дискретных значений $\{d_1, d_2, d_3\}$ в соответствии с решающими правилами:

$$y = \begin{cases} d_1, & \text{если } x_2 < \frac{14.6}{2.25 + (x_1 - 6.5)^2} \\ d_2, & \text{если } \frac{14.6}{2.25 + (x_1 - 6.5)^2} < x_2 < 2.2 \cdot \sqrt{x_1} + 3 \\ d_3, & \text{если } x_2 > 2.2 \cdot \sqrt{x_1} + 3 \end{cases} \quad (9)$$

$$(10)$$

Обучающие и тестирующие выборки показаны на рис. 4. Они содержат 80 и 5000 пар «входы — выход», соответственно. В выборках значения входов выбирались случайно, а значения выхода рассчитывались по (10). Файлы данных доступны в разделе **Fuzzy Logic Toolbox** сайта www.MATLAB.ru.

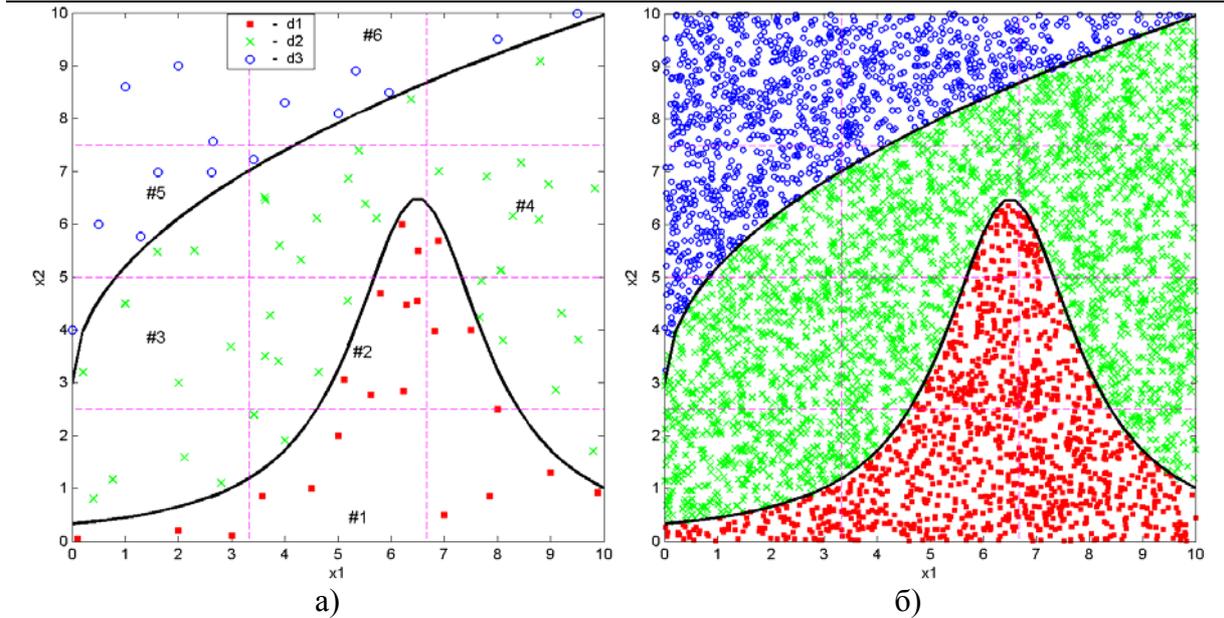


Рис. 4. Обучающая (а) и тестирующая (б) выборки.

На основе этих выборок спроектируем нечеткий классификатор. Входы нечеткого классификатора будем рассматривать как лингвистические переменные, значения которых определяются из следующих термножеств: {«low», «average», «high»} для x_1 , и {«low», «lower average», «higher average», «high»} для x_2 . Формализацию термов осуществим симметричной гауссовской функции принадлежности:

$$\mu(x) = e^{-\frac{(x-h)^2}{2c^2}},$$

где x — элемент универсального множества; h и c — параметры функции принадлежности - координата максимума коэффициент концентрации.

До настройки коэффициенты концентраций всех функций принадлежности равны 2. Координаты максимумов выбирались так, чтобы разбить интервал $[0, 10]$ на три (для x_1) и на четыре (для x_2) равные части (рис. 5а). По рис. 4а эксперт сгенерировал шесть нечетких правил классификации (табл. 1).

Таблица 1.
К примеру 5.2: нечеткая база знаний.

x_1	x_2	y	w (до настройки)	w (классификатор I)	w (классификатор II)	w (классификатор III)
average	low	d_1	1	0.62	0.75	0.71
average	below average	d_1	1	0.41	0.39	0.49
low	below average	d_2	1	0.81	1	0.90
high	higher average	d_2	1	0.46	1	0.71
low	higher average	d_3	1	0.66	0.49	0.65
average	high	d_3	1	0.02	0.02	0.91

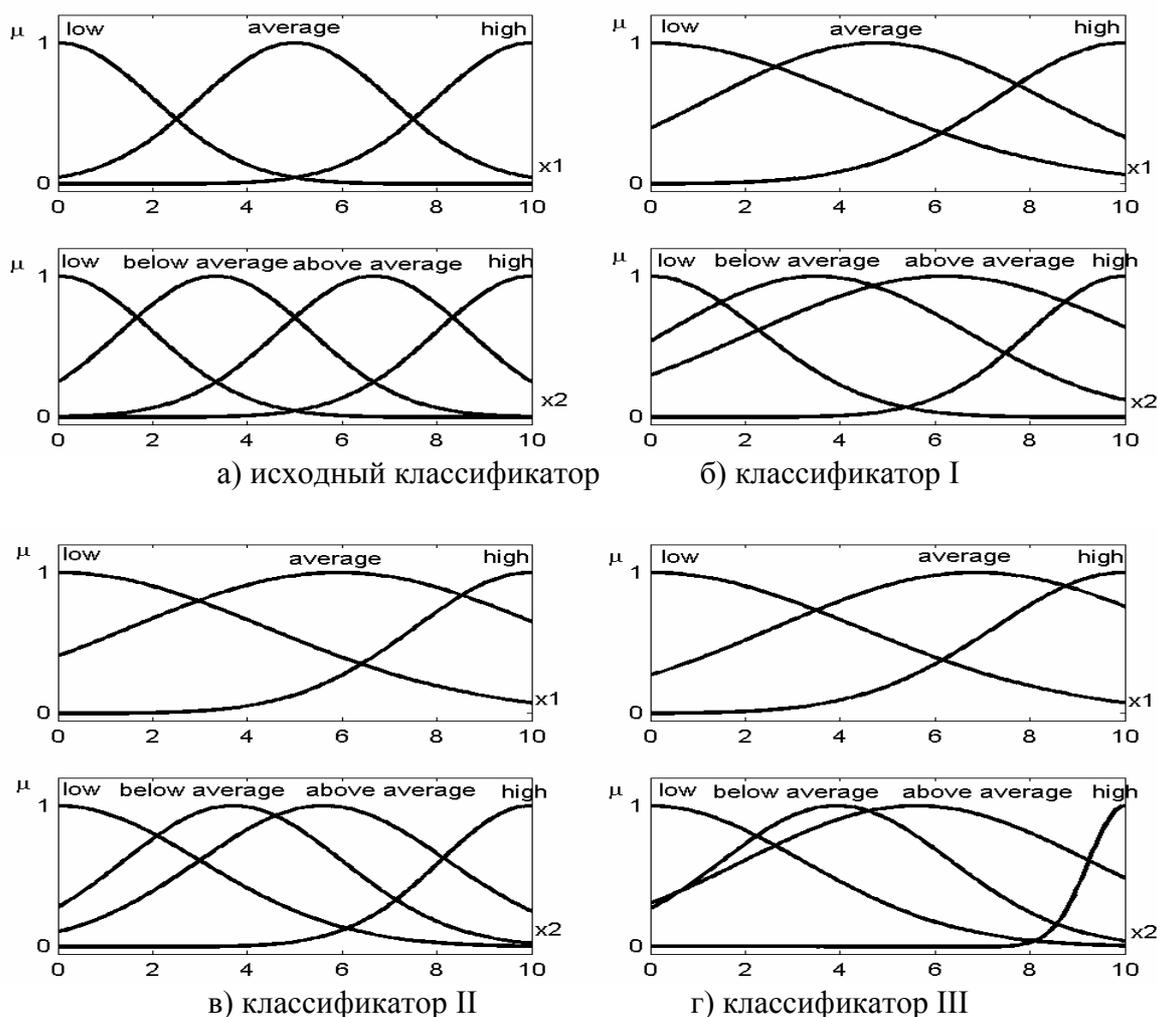


Рис. 5. Функции принадлежности.

Исходная нечеткая база знаний с шестью правилами грубо отражает нелинейные разделяющие кривые - на тестирующей выборке ошибочно классифицировано 26.6% объектов (рис. 6а). После настройки весов правил количество ошибок уменьшилось до 15%, однако безошибочность нечеткого классификатора остается на уровне следующего простого дерева решений:

Если $((x1 > 1.2929) \& (x2 \leq 1))$	то $y = d1$,
Если $((x1 > 4.6335) \& (x2 > 1) \& (x1 \leq 7.5) \& (x2 \leq 6))$	то $y = d1$,
Если $((x2 > 8.3607) \& (x1 > 5.3301))$	то $y = d3$,
Если $((x1 \leq 5.3301) \& (x2 > 6.9107))$	то $y = d3$,
Если $((x1 \leq 1.2929) \& (x2 > 3.4988) \& (x2 \leq 6.9107))$	то $y = d3$,
Иначе,	$y = d2$.

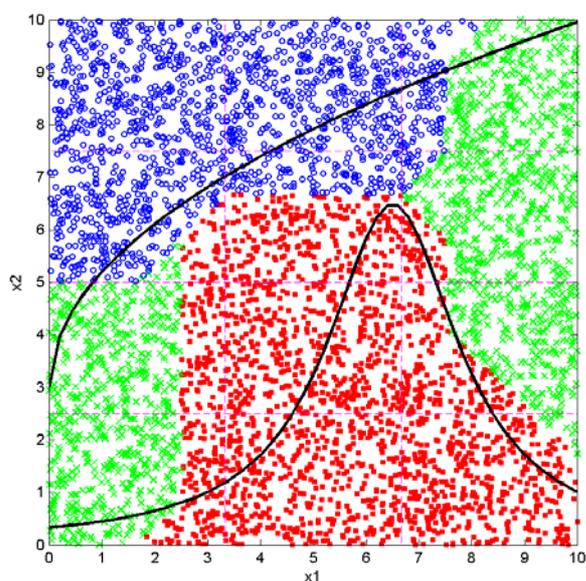
Низкая безошибочность нечеткого классификатора после настройки весов правил объясняется «плохими» функциями принадлежности. Поэтому, необходимо модифицировать не только веса правил, но и функции принадлежности. Будем настраивать следующие 16 параметров нечеткого классификатора:

- 3 координаты максимумов функций принадлежности термов «average», «lower average» и «higher average»;
- 7 коэффициентов концентраций функций принадлежности термов входных переменных;
- 6 весовых коэффициентов правил базы знаний.

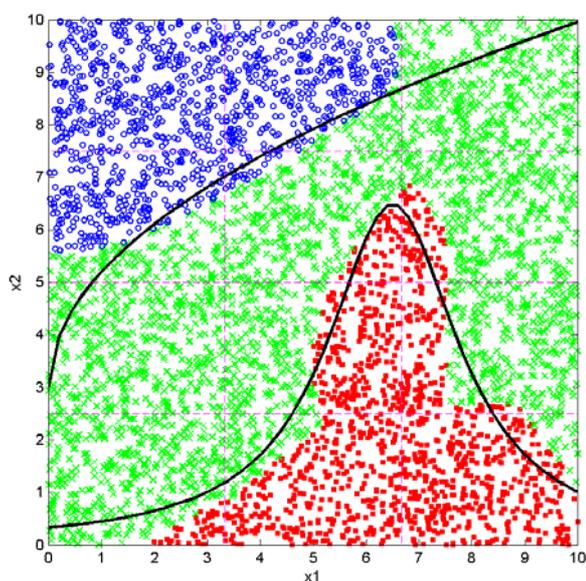
Результаты настройки с использованием различных критериев приведены на рис. 5 и в табл. 1. Использовались такие критерии настройки: критерий I — формула (5); критерий II — формула (7); критерий III - формула (8). Результаты тестирования классификаторов сведены в табл. 2. Классификатор I настроен по критерию I, классификатор II — по критерию II и классификатор III - по критерию III.

Таблица 2.
Результаты тестирования классификаторов.

Классификатор	Критерий I	Критерий II	Критерий III	Безошибочность на тестирующей выборке
Исходный	33.75%	0.433	3.99	26.6%
Классификатор I	6.25%	0.481	1.08	9.78%
Классификатор II	18.75%	0.417	1.49	16.42%
Классификатор III	6.25%	0.416	0.98	9.28%
Дерево решений	7.5%	-	-	15.24%



а) исходный классификатор



б) классификатор I

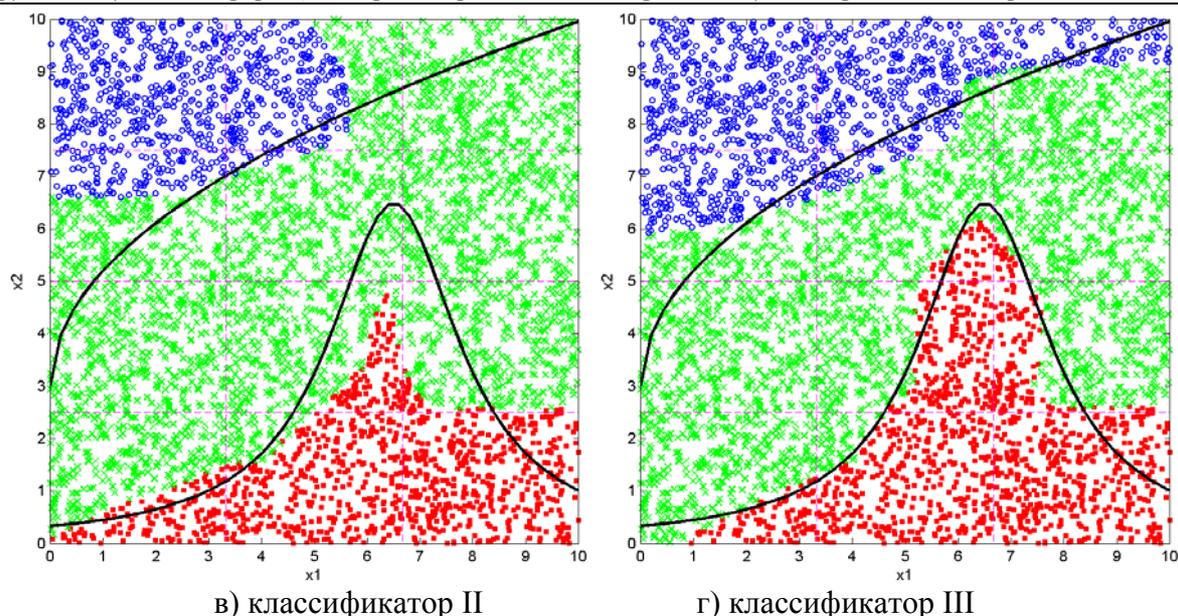


Рис. 6. Классификация на тестирующей выборке.

Выводы

В статье показано как расширить **Fuzzy Logic Toolbox** для проектирования нечетких классификаторов, т. е. систем нечеткого логического вывода с дискретным выходом. Проанализированы критерии настройки нечеткого классификатора. Предложен пакет программ FALEFC для быстрой настройки весов правил нечеткого классификатора. Приведенные примеры проектирования нечетких классификаторов подтверждают эффективность предложенных программ расширения пакета **Fuzzy Logic Toolbox**.

Литература

1. *Kasabov N.* Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering. The MIT Press.— 1996.— 550 p.
2. *Rotshtein A.* Desig and Tuning of Fuzzy Rule-Based System for Medical Diagnosis. In Fuzzy and Neuro-Fuzzy Systems in Medicine (Teodorescu N.H. (ed.)). CRC-Press, 1998.— P.243–289.
3. *Ротштейн А. П.* Интеллектуальные технологии идентификации: нечеткая логика, генетические алгоритмы, нейронные сети.— Винница: УНІВЕРСУМ-Вінниця, 1999.— 320 с.
4. Fuzzy Logic Toolbox. User's Guide, Version 2.— The MathWorks, Inc., 2001.
5. *Штовба С. Д.* Идентификация нелинейных зависимостей с помощью нечеткого логического вывода в системе MATLAB // Exponenta Pro. Математика в приложениях.— 2003.— №2.— С.9–15.

Приложение 1. Листинг функции *fuzzy_classifier*

```

function [decision, mf_grades]=fuzzy_classifier(x, fis, type)
%FUZZY_CLASSIFIER выполняет нечеткий логический вывод для задач
%      классификации.
% x - входной вектор (признаки классифицируемого объекта);
% fis - система нечеткого логического вывода;
% type - формат решения:
%'number' - порядковый номер класса (значение по умолчанию);
%'name' - наименование класса;
% decision - результат классификации объекта x;
% mf_grades - вектор степеней принадлежности каждому классу.
%Требуемые программные средства: Fuzzy Logic Toolbox v.2.X

%Serhiy D. Shtovba shtovba@ksu.vstu.vinnica.ua
%Vinnitsa State Technical University
%$Revision: 1.6 $ $Date: 2004/03/12

%Проверка входных аргументов:
if nargin<2
    error('Необходимо задать 2 или 3 входных аргумента');
end
[tmp1 tmp2]=size(x);
if tmp1~=1
    error('Классификация выполняется только одного объекта');
end
if isfis(fis)==0
    error('Второй аргумент должен быть системой нечеткого вывода');
end
if fis.type(1:6)~='sugeno'
    error('Система нечеткого вывода должна быть типа Сугено');
end
if nargin==2
    type='number';      % <---- Установка значения по умолчанию
end

number_of_decisions=length(fis.output(1).mf);
number_of_rules=length(fis.rule);
[a,b,c,d]=evalfis(x,fis); % <---- Нечеткий логический вывод
%Номера правил с максимальной степенью выполнения:
rule_num_with_max_fulfilment=find(d==max(d));
%количество таких правил:
number_rules_with_max=length(rule_num_with_max_fulfilment);
if number_rules_with_max>1
    %Если таких правил несколько, то возможно объект попал на границу классов:
    colision_flag=zeros(1,number_of_decisions);
    for i=1:number_rules_with_max
        index=rule_num_with_max_fulfilment(i);
        %Помечаем решения с максимальной степенью принадлежности:
        colision_flag (fis.rule(index).consequent )=1;
    end
end

```

```
end
%Суммируем степени принадлежности проблемного объекта по разным прави-
лам:
sum_mf=zeros(1,number_of_decisions);
for i=1:number_of_rules
    index=fis.rule(i).consequent;
    sum_mf(index)=sum_mf(index)+d(i);
end
%оставляем классы с максимальной степенью принадлежности (через *)
%и выбираем класс по максимальной сумме степеней принадлежности:
[tmp1 number_of_the_class]=max(sum_mf.*colision_flag);
else number_of_the_class=fis.rule(rule_num_with_max_fulfilment).consequent;
end
%Возврат результата классификации в требуемом формате:
switch type
case 'number', decision=number_of_the_class;
case 'name', decision=fis.output.mf(number_of_the_class).name;
otherwise, error('Допустимые значения для 3-го аргумента: number или name')
end
if nargin==2
    mf_grades( 1:number_of_decisions )=0;
    for i=1:number_of_rules
        index=fis.rule(i).consequent; %номер класса в i-м правиле
        %Объединим через операцию максимума степени принадлежности одному и
        %тому же классу по различным правилам:
        mf_grades(index)=max(mf_grades(index), d(i));
    end
end
end
```

Приложение 2. Листинг сценария настройки нечеткого классификатора ирисов

```
%Настройки нечеткого классификатора ирисов
%Требуемые программные средства: Fuzzy Logic Toolbox v.2.X
% Optimization Toolbox v.2.X
% FALEFC v.1.0
%Serhiy D. Shtovba shtovba@ksu.vstu.vinnica.ua
%Vinnitsa State Technical University
%$Revision: 1.2 $ $Date: 2004/03/14

%Загрузка исходной нечеткой модели:
fis=readfis('iris_сд.fis');
RULE_order=rule_order_fuz_cl(fis);
%Загрузка обучающей выборки:
[INP OUT_c OUT_mu]=dp_for_fuzzy_cl_learning(fis, 'iris.dat');
%Расчет RMG:
num_rule=length(fis.rule);
for i=1:num_rule
    fis.rule(i).weight=1;
end
```

```

RMG=rmg_fuzzy_cl(INP, fis);
%НАСТРАИВАЕМЫЕ
ПАРАМЕТРЫ
vlb_w(1:num_rule)=0;           %нижняя граница
w0(1:num_rule)=rand(1, num_rule); %начальная точка
vub_w(1:num_rule)=1;           %верхняя граница
%-----
%ПАРАМЕТРЫ ОПТИМИЗАЦИИ:
options=[];
options=optimset('Display', 'iter');
options.DiffMinChange=0.1;
options.DiffMaxChange=0.25;
options.LargeScale='off';
options.MaxIter=20;
options.MaxFunEvals=150;
%Формирование обучающей выборки:
ii=1;
for i=1:150
    if rem(i,5)~=0
        tr_index(ii)=i; ii=ii+1;
    end
end
%Оптимизация:
[wopt, delta]=fmincon(@ob_fun_fast_w, w0, [], [], [], [], vlb_w, ...
    vub_w, [], options, RULE_order, RMG(tr_index,:), OUT_c(tr_index,:), ...
    3, OUT_mu(tr_index,:))
%Тестирование по трем критериям на обучающей выборке:
[decision, delta_m, delta_s, delta_p] =fast_cl_testing_with_rmg(wopt, ...
    RULE_order, RMG(tr_index,:), OUT_c(tr_index,:), OUT_mu(tr_index,:));
%Тестирование по трем критериям на всей выборке:
[decision, delta_m, delta_s, delta_p] =fast_cl_testing_with_rmg(wopt, ...
    RULE_order, RMG, OUT_c, OUT_mu);
%Запись оптимальных весов в систему нечеткого вывода:
for i=1:num_rule
    fis.rule(i).weight=wopt(i);
end

```

УДК 519.6

НЕЙРОННЫЕ СЕТИ ТРЕТЬЕГО ПОКОЛЕНИЯ. РАСПОЗНАВАНИЕ ОБРАЗОВ И СКРЫТЫХ ЗАВИСИМОСТЕЙ В ПРОИЗВОЛЬНОМ СИГНАЛЕ С ПОМОЩЬЮ ИМПУЛЬСНЫХ НЕЙРОСЕТЕЙ С ИСПОЛЬЗОВАНИЕМ СРЕДЫ MATLAB

Юзбашев Д. А.

Московский государственный университет им. М.В.Ломоносова,
факультет вычислительной математики и кибернетики, Москва,
e-mail: dimuse@mail.ru

Введение

Проводя классификацию нейронных сетей в соответствии с их базовыми вычислительными единицами — нейронами, можно достаточно четко разделить все виды нейросетей на три поколения. Первое поколение сетей представлено персептронами и пороговыми нейросетевыми архитектурами. Они восходят к таким моделям нейронных сетей, как многослойные персептроны (Threshold circuits), сети Хопфилда и Машины Больцмана. Характерной особенностью для моделей данного поколения является то, что они могут только выдавать дискретные величины на выходе. На самом деле, они являются *универсальными* для вычислений с дискретными (цифровыми) входом и выходом, и любая булева функция может быть вычислена с помощью некоторого многослойного персептрона с одним скрытым слоем.

Второе поколение нейросетей базируется на нейронах (вычислительных единицах), которые применяют взвешенное суммирование входов (или полиномиальное) — функцию активации, с непрерывно-определенным множеством возможных выходных значений, такие как *сигмоидная функция* $\sigma(y) = (1 + e^{-y})^{-1}$ или *линейная функция насыщения* π :

$$\pi(y) = \begin{cases} 0, & \text{при } y < 0 \\ y, & \text{при } 0 \leq y \leq 1 \\ 1, & \text{при } y > 1 \end{cases}$$

Типичными примерами нейросетей этого поколения являются сети прямого распространения, рекуррентные сети, а также сети нейронов с радиально-базисными функциями. Такие сети также могут вычислять (с помощью округления значений на выходах сети) произвольные булевы функции. Было показано, что вычислять некоторые булевы функции нейронные сети второго поколения могут с *меньшим* числом входов, чем сети

первого поколения. Кроме того, нейросети второго поколения могут вычислять функции с аналоговыми входом и выходом. Действительно, эти сети являются универсальными для аналоговых вычислений в том смысле, что любая непрерывная функция, определенная на компакте, может быть аппроксимирована достаточно хорошо (в смысле равномерной сходимости, то есть в нормированном бесконечномерном пространстве интегрируемых по Лебегу функций) с помощью нейросети такого типа (второго поколения) с одним скрытым слоем. Еще одной особенностью этих моделей нейросетей является то, что они поддерживают градиентные методы обучения, такие как обратное распространение ошибки.

С точки зрения биологической интерпретации, в нейросетях второго поколения выходы сигмоидных нейронов представляют (собой) текущий уровень возбужденности биологического нейрона. Так как известно, что биологический нейрон способен переходить в состояние возбуждения на различных промежуточных частотах между минимальной и максимальной частотами, модели нейронов второго поколения более реалистичны, нежели «первые» нейроны, с точки зрения «интерпретации уровня (частоты) возбуждения». Однако, по крайней мере, по сравнению с *быстрыми* аналоговыми вычислениями, выполняемыми сетями нейронов, например, коры головного мозга, вопрос об «интерпретации эффекта возбуждения» для искусственных нейронов остается открытым. Нейробиологами было установлено, что анализ визуальных образцов и их классификаций может быть выполнена человеком всего за 100 мсек. Более того, только визуальная обработка нейронами коры головного мозга осуществляется за 20-30 мсек. С другой стороны, частота возбуждений нейронов, вовлеченных в этот процесс обычно не превышает 100 Гц., и следовательно, по крайней мере 20-30 мсек понадобится только для того, чтобы измерить уровень возбуждения нейрона. Таким образом, кодирование аналоговых переменных частотой возбудимости представляется весьма сомнительным в контексте скорости обработки информации нейронами коры мозга.

С другой стороны, как было показано экспериментально еще в середине 90-х, многие биологические системы используют временные замеры потенциалов действий («спайков») для кодирования информации. Эти нейробиологические выводы легли в основу исследований и формирования третьего поколения моделей нейронных сетей с использованием так называемых *спайк-нейронов* (или *импульсных* нейронов) в качестве вычислительной единицы. С практической точки зрения, чипы, кодирующие информацию временными интервалами между импульсами, получили в последнее время широкое распространение в связи с преимуществом этого способа кодирования по сравнению с другими (традиционными) методами, используемыми в hardware.

Не вдаваясь в детали всех электрохимических процессов, проходящих в сетях биологических нейронов, следует отметить, что искусствен-

ные импульсные нейроны — это некоторое «приближение» биологических, так как они сфокусированы на некоторых конкретных принципах в процессах последних. Однако, по сравнению с нейронами первых двух, нейроны третьего поколения — значительно более реалистичны. В частности, они лучше описывают фактический выход биологического нейрона, и следовательно, они позволяют нам исследовать возможность использования времени (интервалов) как средство вычислений и связи. Индивидуальные временные интервалы спайк-нейронов играют ключевую роль в вычислениях всей сети в целом, по сравнению с ролью (связанной с синхронизацией сети) временных лагов в вычислениях в двух предыдущих поколениях сетей. К тому же, выход импульсного нейрона состоит из множества точек во времени, когда этот нейрон «отвечал» приходу импульса.

Классический импульсный нейрон

Возбудимость нейрона — это его способность отвечать на синаптическое воздействие потенциалом действия. В самой простой модели импульсного нейрона предполагается, что нейрон возбуждается в случае, если его мембранный потенциал P достигает определенного порога θ . Величина мембранного потенциала является основным параметром, который определяет значения важнейших показателей функционального состояния нейрона — его возбудимость и лабильность [2].

Потенциал P это сумма так называемых возбуждающего постсинаптического потенциала (ВПСП) и тормозного постсинаптического потенциала (ТПСП), которые являются результатами возбуждения других нейронов, которые подсоединены к данному синаптическими связями. Это называется интегративной функцией нейрона: общее изменение мембранного потенциала нейрона (МП) является результатом сложного взаимодействия (интеграции) местных ВПСП и ТПСП всех многочисленных активированных синапсов на теле и дендритах клетки. На мембране нейрона происходит процесс алгебраического суммирования положительных и отрицательных колебаний потенциала (возникающих вследствие приходов импульсов от других нейронов). При одновременной активации нескольких возбуждающих синапсов общий ВПСП нейрона представляет собой сумму отдельных местных ВПСП каждого синапса. При одновременном возникновении двух различных синаптических влияний — ВПСП и ТПСП — происходит взаимное вычитание их эффектов. Ответный разряд нейрона (потенциал действия) возникает лишь тогда, когда изменения мембранного потенциала достигают порогового значения — *критического уровня деполяризации* (это название из нейрофизиологии — когда положительные и отрицательные заряды поляризуются на стенках мембраны, а при ВПСП-или ТПСП-процессах происходят их соответственно деполяризация или гиперполяризация, так как в случае ВПСП в клетку проникают положи-

тельные заряды ионов натрия, а при ТПСР — ионов калия и хлора из клетки). Тип синаптической связи в некоторых моделях искусственных импульсных нейронов (в частности, которые мы будем рассматривать) задается знаком соответствующего входного веса, который может изменяться в процессе обучения. Однако существуют и другие модели ([1]), в которых все веса предполагаются $\omega_{u,v} \geq 0$ (где u — это пресинаптический нейрон для данного нейрона v), а тип связи жестко задается видом *ответной функцией* $\varepsilon_{u,v}(t-s)$, где знак (т. е. вид) определяется типом синаптической связи $u \leftrightarrow v$, s — это момент времени генерации нейроном u потенциала действия (его возбуждением), а t — это время расчета значения мембранного потенциала P_v нейрона v , который определяется формулой $\omega_{u,v}(t)\varepsilon_{u,v}(t-s)$. Вес $\omega_{u,v}$ характеризует силу синаптической связи между нейронами $u \leftrightarrow v$. В контексте *обучения* сети вес должен представляться функцией времени $\omega_{u,v}(t)$, так как стремительные изменения весов являются существенной деталью для вычислений в системах биологических нейронов.

На рис. 1 приведен один из возможных видов функций $\varepsilon_{u,v}(t-s)$ [1].

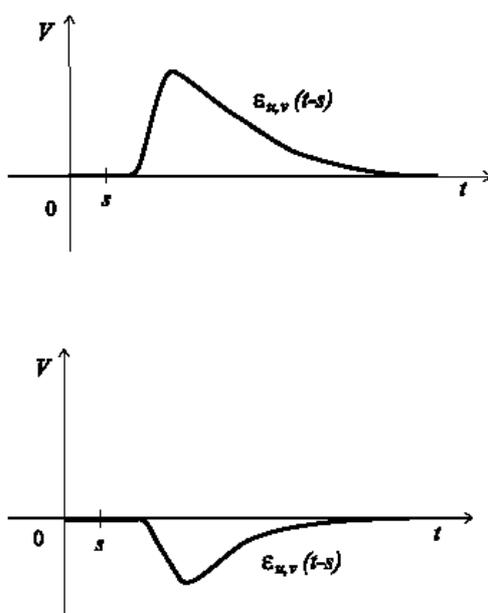


Рис. 1. Типичная форма ответной функции (ВПСП и ТПСР) для биологического нейрона.

Следует отметить, что, вообще говоря, вид функции $\varepsilon_{u,v}(t-s)$ должен зависеть от некоторых внутренних параметров (время задержки на передачу импульса через пресинапс; параметры, характеризующие крутизну подъема и пологость ската в период после передачи-прихода импульса), так как, возможно, их следует сделать адаптивными.

Все рефракторные эффекты нейрона (абсолютная/относительная рефракторные фазы) моделируются подходящей *пороговой функцией* $\theta_v(t-t')$, где t' — это время последней генерации спайка. В детерминистских моделях спайк-нейронов предполагается, что нейрон v возбуждается как

только $P_v(t)$ пересекает снизу функцию $\theta_v(t-t')$ (здесь предполагается, что потенциал $P_v(t)$ нейрона равен 0 в отсутствии постсинаптического потенциала, а пороговое значение $\theta_v > 0$, так как это - математически удобнее, — см. рис. 2) [1].

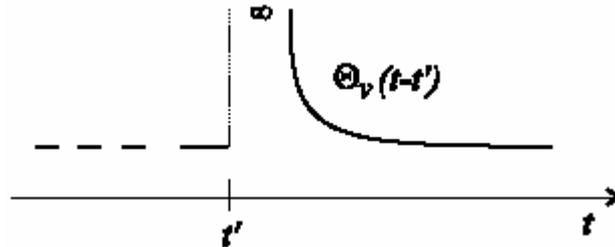


Рис. 1. Типичная форма пороговой функции биологического нейрона.

Прежде чем мы перейдем к более детальному рассмотрению формального строения спайк-нейрона, отметим, что для того, чтобы раздражитель (собирательное понятие: степень раздражающего воздействия источников импульсов) мог вызвать у данного нейрона возбуждение, он должен быть достаточно сильным — пороговым или выше порогового —, и достаточно длительным — в зависимости от времени меняется сила раздражения. Последнее говорит о том, что зависимость пороговой силы раздражителя от длительности его действия носит характер обратной зависимости (гипербола) — чем меньше по времени действует на ткань раздражитель, тем выше требуется его сила для инициации возбуждения (см. рис. 3)[3].

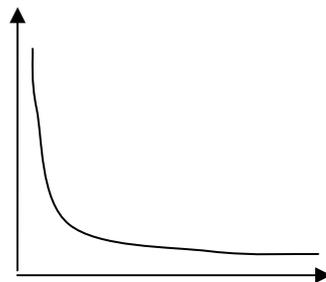


Рис. 3. Типичная форма пороговой функции биологического нейрона.

Формальное описание сети

Сеть импульсных нейронов состоит из конечного множества V спайк-нейронов, множества $E \subseteq V \times V$ синапсов, весов $\omega_{u,v}(t)$, ответной функции $\varepsilon_{u,v}(t-s): R_+ \rightarrow R$ для каждого синапса $\langle u,v \rangle$ из E , и пороговой функции $\theta_v: R_+ \rightarrow R_+$ для каждого нейрона v из V .

Если обозначить $F_u \subseteq R_+$ - множество моментов времен генерации импульсов нейроном u , тогда потенциал на мембране нейрона v во время t вычисляется следующим образом:

$$P_v(t) := \sum_{u: \langle u,v \rangle \in E} \sum_{s \in F_u: s < t} \omega_{u,v}(t) \varepsilon_{u,v}(t-s)$$

В детерминистских моделях (отсутствует шум) нейрон v вырабатывает ответный импульс как только значение $P_v(t)$ достигает порога $\theta_v(t-t')$, где t' — время последнего возбуждения нейрона v .

Предполагается, что для некоторого подмножества V_{in} из V — входных нейронов — множество моментов генерации импульсов F_u нейронами u из V_{in} определяется не данной формулой, а задается извне (подразумевается, что входные нейроны подсоединены к некоторым рецепторам, от которых они и получают в некоторые моменты времени импульсы). Для всех остальных нейронов вычисление уровня потенциала и моментов времени возбуждения определяется по вышеуказанному правилу. В частности, выход всей сети задается в форме временных замеров F_v для нейронов v некоторого множества *выходных нейронов* V_{out} из V .

Экспериментально было установлено, что для разных живых биологических нейронов возбуждение наступает с различными задержками в ответ на последовательности атак на них текущими воздействиями. Поэтому рассматриваются также так называемые стохастические или зашумленные виды моделей спайк-нейронов, в которых разность $P_v(t) - \theta_v(t-t')$ определяет *вероятность* того, что нейрон v сгенерирует ответный импульс в момент времени t . Выбор же точных моментов времени возбуждения остается за некоторым стохастическим процессом, и может так оказаться, что нейрон v не пришел в состояние возбуждения за некоторый период I , в течение которого $P_v(t) - \theta_v(t-t') > 0$, или, наоборот, нейрон сгенерировал ответный разряд в тот момент t , когда $P_v(t) - \theta_v(t-t') < 0$.

Последняя описанная модель импульсного нейрона является базовым прототипом большинства математических моделей сетей спайк-нейронов. Различные модификации и различия могут существовать между этими моделями относительно рефракторных эффектов и сброса мембранного потенциала сразу после генерации импульса.

Импульсные ансамблевые нейросети

Теперь перейдем к рассмотрению еще одного интересного прототипа нейросети импульсных нейронов, используемого для нахождения скрытых закономерностей и образов в неструктурированном потоке информации.

Как было сказано во введении, кодирование информации временными интервалами между импульсами является достаточно распространенным и эффективным. Таким образом, любой структурный сигнал может быть представлен набором времен генерации импульсов некоторого множества нейронов, при условии, что до начала подачи структурного сигнала сеть была приведена в некоторое состояние стабилизации на чистом шуме. То есть любая полезная информация в сигнале может быть отображена в терминах возбуждения некоторого набора нейронов, отреагировавших именно на данное событие во входном сигнале, так, что различные собы-

тия (структурные элементы сигнала) кодировались бы разными наборами составляющих нейронов. Тогда обученная нейросеть будет представлять собой ансамблевую сеть, в которой ответом на уникальное событие в потоке будет активизация уникальной группы нейронов.

Так как произвольный сенсорный сигнал представляет собой довольно сложную пространственно-временную структуру, возникает задача распознавания образов (выделения закономерностей) при обучении «без учителя» (unsupervised learning), так как обучение в данном случае представляет собой обобщение правила Хебба нейрональной синаптической пластичности [4].

Задача распознавания любой (!) пространственно-временной структуры декомпозируется в представление о «базовом» уровне распознавания образов в сигнале и более высоких уровнях (слоев) нейросети в зависимости от типа распознаваемого сигнала.

Описание базового уровня модели сети

Базовый уровень представляет собой набор нейронов, присоединенных непосредственно к рецепторам, которые распознают низкоуровневые закономерности типа причинно-следственных связей [4]. Как уже было сказано, такие структуры во входном зашумленном сигнале представляются сетью в виде реакции на них определенных нейронов, объединенных в ансамбль для идентификации *именно* этой структуры. Так как а priori неизвестны свойства сигнала, количество нейронов базового уровня делается заведомо достаточно большим (несколько сотен и более). Кроме того, параметры нейронов при их инициализации задаются неодинаково: все множество нейронов разбивается на несколько непересекающихся групп со сходными (одинаковыми) параметрическими свойствами, такими как длина рефракторных фаз (как абсолютной, так и относительной), пороговое значение, параметр регуляции асимметрии модификации весов нейронов [4] и т. д..

В модели [4] было использовано еще несколько интересных элементов в работе сети, среди которых — *стабильность* нейрона и *подавление (блокирование)* нейронов в рамках одной группы (что, кстати, отражает взаимодействие биологических нейронов некоторых тканей, такое, как непостоянность способности синапсов передавать нервные импульсы [2]). Стабильность — это важный фактор, который характеризует меру обучения нейрона. Так как задачей каждого нейрона является реакция (как активная, так и пассивная) на каждый проходящий от рецепторов паттерн (набор импульсов с сенсоров) в составе с другими нейронами (собственно, этот состав и «распознает» данный паттерн), которая определяется конкретной комбинацией его весов, необходимо было разработать механизм, который бы предотвращал (с вероятностью, обратной величине стабильно-

сти) резкое изменение значений весов, в случае устойчивости данной реакции нейрона, что привело бы к «потере состояния обученности». Аппарат стабильности, предложенный Киселевым М.В. в ([4]), представляет собой описание динамики весов нейронов дифференциальными уравнениями второго порядка, тогда как в классической модели это было представлено уравнениями первого порядка. Таким образом, чем нейрон «умнее», тем выше его стабильность (возможность сохранить свои «знания») и, тем самым, - выше устойчивость в ответ на любые влияния.

Вторым важным механизмом работы сети является подавление нейронов внутри одной группы. Введение этого принципа функциональности связано с тем, что изначально, перед тем как подавать на рецепторы сигнал, необходимо привести сеть в некоторое равновесное устойчивое состояние путем подачи чистого шума на сенсоры. Но тогда, в случае прихода структурного сигнала, нейроны будут одновременно реагировать на появление паттерна, и вся сеть научится распознавать только такой образ. Это неправильно. Во избежание этого эффекта и был реализован следующий механизм [4]. В рамках одной группы (инициация групп нейронов происходит до начала работы алгоритма) каждый нейрон может блокировать на некоторое время работу других нейронов (то есть увеличивать время, в течение которого заблокированный нейрон не может сгенерировать ответный импульс). Таким образом, реакцию на конкретный пришедший паттерн не могут вырабатывать «соседние» нейроны со сходными значениями параметров. Так обеспечивается возможность нейросети распознавать многие различные паттерны.

Адаптивная настройка весов всех нейронов с использованием вышеуказанных механизмов позволяет реализовать самообучение сети. В частности, изначально в системе нет указания на тип синаптической связи между рецепторами и входами нейронов, поэтому формирование возбуждающих и тормозных эффектов происходит online: все веса нейронов могут принимать значения от -1 до $+1$, и, так как порог возбудимости — величина положительная (тоже самонастраивающаяся для каждого нейрона), то, соответственно, знак веса характеризует тип его синаптической связи, а ключевым фактором здесь является переменная величина «асимметрии вознаграждения» (REWARD_ASSYMETRY) [4], которая совершает случайное блуждание с шагом, зависящим от стабильности всего нейрона.

Следующие уровни модели

Дальнейшая работа нейросети, а, точнее, ее более высоких уровней, уже связана не с отдельными нейронами, а происходит в терминах ансамблей. Например, в задаче распознавания речи, формирующиеся ансамбли нейронов могут (и, по всей видимости, — должны) представлять отдельные морфемы (как структурная единица для более высоких уровней). И на

более высоких уровнях уже могут применяться различные алгоритмы и методы обработки «ансамблевой» информации (такие, как обучение с учителем, например).

Понятно, что данный алгоритм подразумевает реализацию в виде *electronic hardware*, так как требует очень больших мощностей, и, как уже было сказано во введении, электронное кодирование информации импульсными задержками представляется очень эффективным для высокотехнологических задач.

Заключение. Использование среды разработки MATLAB.

Для исследования и дальнейшей разработки механизма искусственных ансамблевых нейронных сетей для распознавания скрытых образов используются симуляционные возможности среды разработки *matlab*, как возможный подход к созданию имитационной модели сети. Планируется создать в среде модель сети спайк-нейронов и проверить ее работоспособность на модельных примерах. В данном случае уместным является использование модуля цифровой обработки сигнала для моделирования чистого шума и экспериментального структурного сигнала (чтобы заранее были известны все свойства, паттерны и скрытые зависимости в его частях). Данный опыт состоит в следующем. Так как на чистом сигнале система приходит в состояние равновесия — никакие образы распознаны не могут быть (их там просто нет). Но когда системе подается на рецепторы структурный сигнал — система выходит из своего устойчивого состояния, так как в таком сигнале уже есть спайки, то есть присутствуют флуктуации, значимо отличающиеся по силе воздействия от тех, которые представлены в шуме. Таким образом, в сети начинают происходить изменения весов некоторых нейронов - увеличиваются значения тех весов, нейроны которых возбудились в ответ на пришедший спайк на сенсорах, и, соответственно, уменьшаются веса остальных нейронов. Одновременно с этим процессом модифицируется значение стабильности каждого конкретного нейрона. Следует заметить, что в разных нейронах все эти действия происходят независимо от других нейронов, но «внутри» отдельного нейрона все процессы модификации сильно связаны, и главным связующим звеном является один из самых важных параметров нейрона — асимметрия вознаграждения. Название связано с тем, что если нейрон отвечает генерацией нового импульса в ответ на входной спайк — это значит, он обучается (или уже обучился) распознавать данный образ в сигнале и получает некоторое вознаграждение — увеличиваются его веса (примечание: конечно же неправильно говорить, что именно нейрон *один* научился распознавать паттерн, так как паттерн распознается группой нейронов, а сам этот нейрон может входить сразу в несколько ансамблей). А если же нейрон пассивно отреагировал на данную структуру в сигнале — его вес будет уменьшен на

некоторую величину (тем самым, «культивируя» в нем реакцию на эту структуру). Но важно то, что увеличение значения веса происходит на большую величину, с чем и связано название параметра.

Далее, если после достаточно долгого периода времени (достаточно длинной истории) происходит стабилизация некоторого высокого уровня стабильности нейронов (скажем, максимальной стабильности) по сравнению с начальным уровнем, то, значит, нейроны обучены распознавать такой сигнал. То есть стабильность нейрона — это не только величина, контролирующая обучения каждого нейрона, но и показатель эффективности работы всей сети в целом.

Если на данном искусственном сигнале сеть будет работать (а, как показывают предварительные опыты, — она распознает структуру сигнала), то это становится поводом для подачи ей на вход любого плохо структурированного сигнала (например, речи), и проведения дальнейших опытов и разработок по модификации структуры сети.

Кроме того, планируется распараллелить данный алгоритм (то есть программу на matlab) и попробовать запустить его на суперкомпьютере, так как в самом начале уже было сказано, что его реализация в electronic hardware (в виде чипа) должна иметь значительные преимущества в способах представления и обработки информации.

Таким образом, запуск системы на высокопроизводительном кластере, где эмерджентное формирование ансамблей сети физически представляется группами процессоров - является одной из очень важных и первостепенных задач исследования.

Литература.

1. *Maas W.* Networks of Spiking Neurons: The Third Generation of Neural Networks Model.— ECCS TR 96-031, 1996.
2. *Шульговский В. В.* Физиология высшей нервной деятельности с основами нейробиологии.— М.: Академия, 2002 г.
3. *Иванов А.* Развитие структуры искусственных нейронных сетей на основе свойств биологических нейронов.— 2003.
4. *Kiselev M. V.* Statistical Approach to Unsupervised Recognition of Spatio-temporal Patterns by Spiking Neurons.— IEEE, 2003.