МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Московский государственный технологический университет «СТАНКИН»

(ФГБОУ ВО МГТУ «СТАНКИН»)

Институт
автоматизации
и робототехники

Кафедра

робототехники и мехатроники

Методические указания к лабораторной работе №5

«Управление исполнительным устройством с одной степенью подвижности с помощью нейронной сети»

Дисциплина: «**Современные методы управления в робототехнике и мехатронике**»

Р.В. Колесниченко, Ю.В. Илюхин

Москва 2017 г.

Цель: научиться управлять исполнительным устройством с одной степенью подвижности с помощью нейросетевого регулятора. Изучить архитектуру нейронной сети регулятора на основе эталонной модели (Model Reference Controller).

Задачи: создать динамическую модель исполнительного устройства с одной степенью подвижности, создать динамическую модель эталонной модели. Также создать нейронную сеть, решающую задачу идентификации объекта управления и нейронную сеть, решающую задачу управления динамической моделью однозвенного робота. Обучить нейронную сеть на управление таким объектом.

Постановка задачи.

Исполнительное устройство с одной степенью подвижности имеет параметры, представленные в таблице 1.

Таблица 1. Параметры исполнительного устройства.

параметр	описание параметра	значение	размерность
b	коэффициент вязкого трения	2	кг/м/с
m	масса звена робота	1	кг
l	длина звена робота	1	м

В работе принято допущение, что масса звена исполнительного устройства сосредоточена на его конце, как представлено на рисунке 1. Коэффициент вязкого трения пропорционален скорости поворота звена.

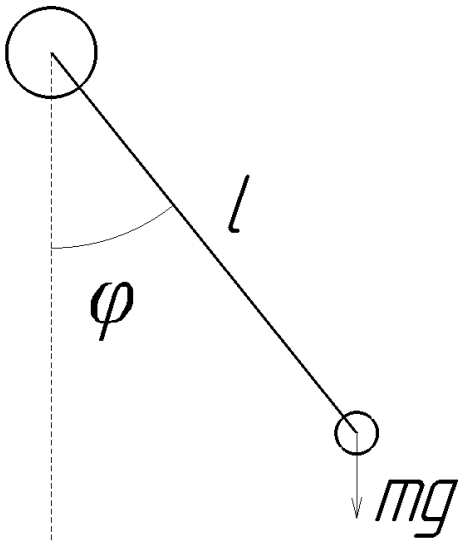


Рисунок 1. Звено исполнительного устройства.

Требования, которым должна удовлетворять система управления звена, представлены в таблице 2.

Таблица 2. Требования к системе управления.

	Уст. ошибка, рад	Отн. перерегулирование,%	Время установления, с
при входных возд-ях до 1 рад	< 0.04	0	2

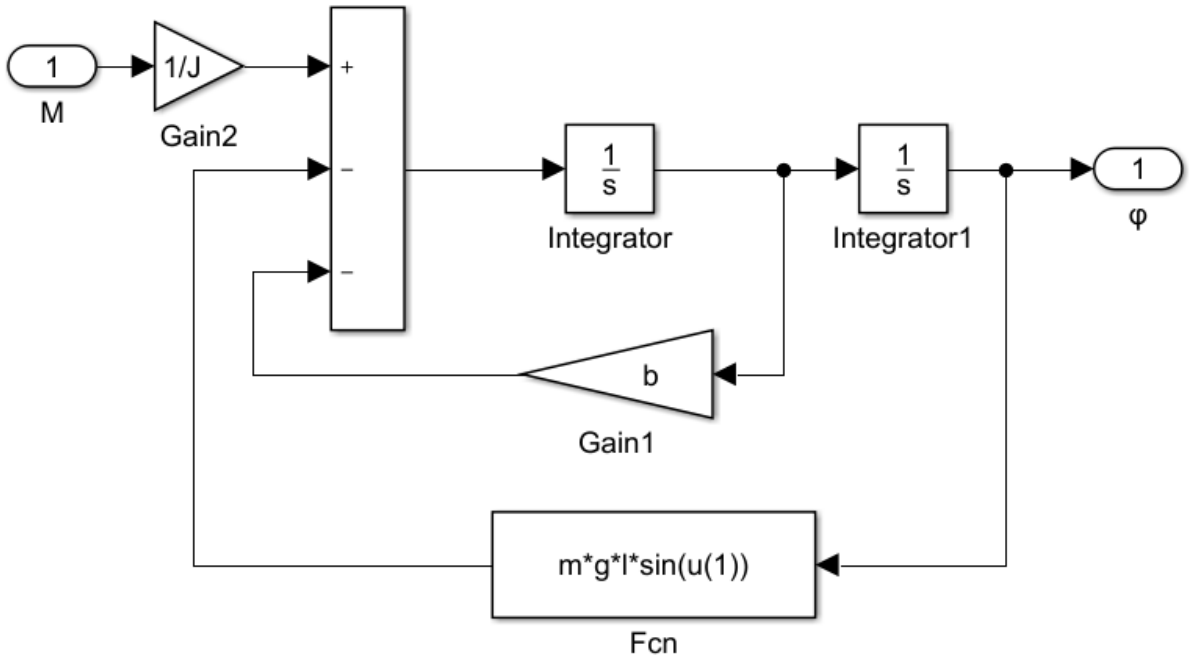
Динамическая модель звена.

Запишем уравнение моментов, действующих на звено:

$$J\varepsilon = -mgl\sin(\varphi) + M - b\omega,$$

где M - момент, действующий на звено со стороны привода.
Запишем уравнение, выразив угловую скорость и ускорение через φ:

$$J\frac{d^2\varphi}{dt^2} = -mgl\sin(\varphi) + M - b\frac{d\varphi}{dt} \quad (1)$$



Эталонная модель звена.

Передаточная функция линеаризованной относительно 0 модели звена следующая:

$$\frac{\varphi(s)}{M(s)} = \frac{\frac{1}{J}}{s^2 + bs + mgl}$$

Обратим внимание, что собственная частота колебательного звена равна $\omega_n = \sqrt{mgl}$. Для удовлетворения требований, представленных в таблице 2, коэффициент затухания системы примем $\xi = 1$. В таком случае передаточная функция эталонной модели будет иметь вид:

$$\frac{\varphi(s)}{M(s)} = \frac{mgl}{s^2 + 2\sqrt{mgl}s + mgl} \quad (2)$$

Архитектура нейронной сети.

Нейронные сети могут быть использованы для решения задач распознавания образов, кластеризации и классификации, для аппроксимации и фильтрации сигналов, а также для управления динамическими процессами. Последним свойственно наличие элементов запаздываний на входах, выходах сети и между её слоями, а также наличие обратных связей. Элементы запаздываний осуществляют запоминание соответствующих последовательностей входа и выхода, а наличие обратной связи в самой сети придаёт ей свойство самоорганизации. Нейроны такой сети могут быть обучены выявлению не только

групп (кластеров) векторов входа, обладающих некоторыми общими свойствами, но и предсказанию будущего поведения нелинейного управляемого процесса. В таком случае становится особенно важным обеспечить эффективность и внутреннюю устойчивость или ограниченность всех переменных. Обратная связь в данном случае подразумевает измерение сигнала на выходе объекта управления, определение разности между измеренными значениями и желаемыми значениями, подаваемыми, например, с эталонной модели, и использование этой разности для определения, какие входы системы вызывают наибольшее изменение этой разности с целью соответствующей подстройки весов в процессе обучения.

На рисунке 2 представлены различные архитектуры нейронных сетей для управления динамическими процессами. Впервые использование нейронных сетей в системах управления было предложено К.С. Нарендрой и К. Пасарати [Narendra & Pasarthy, 1990] и П. Вербосом [Werbos, 1990].

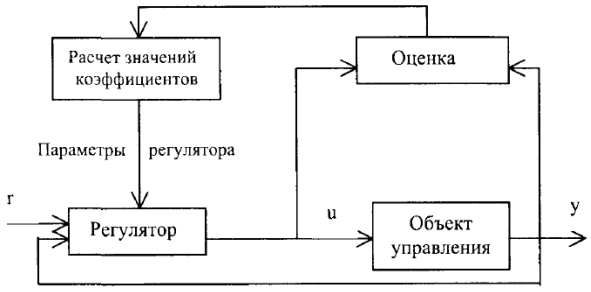


Рисунок 2. Модель с обратной связью с регулируемыми в реальном масштабе времени коэффициентами.

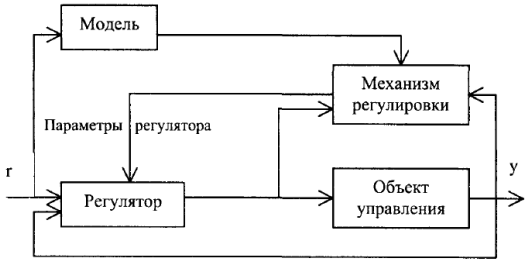


Рисунок 3. Адаптивное управление с эталонной моделью.

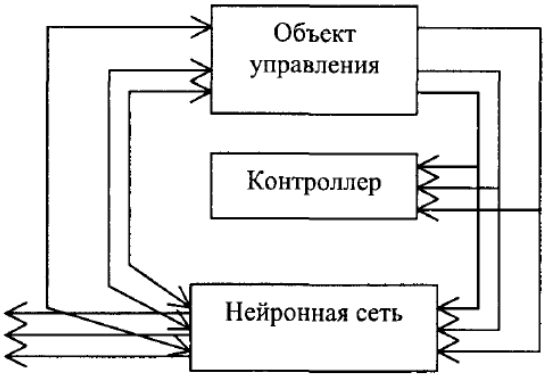


Рисунок 4. Управление, основанное на “копировании” нейронной сетью существующего контроллера.

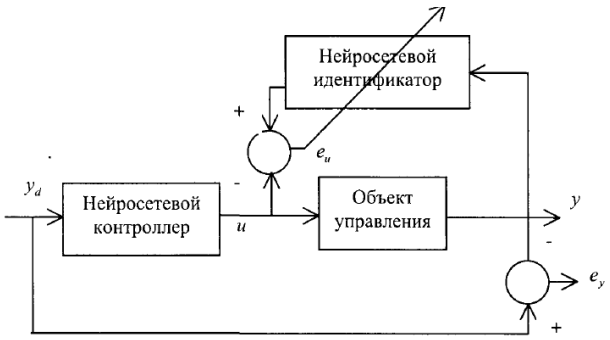


Рисунок 5. Схема косвенного управления.

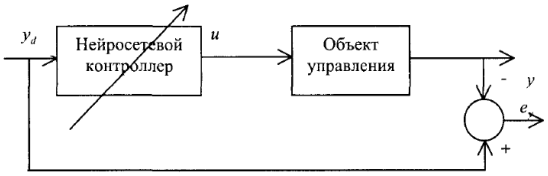


Рисунок 6. Схема прямого управления.

Основными преимуществами нейросетевого управления можно назвать отсутствие требований к линейности системы, эффективность в условиях шумов и отклонений

параметров объекта. Нейросетевое управление по сравнению с другими методами имеет наилучшую робастность рассогласования объекта управления и одну из лучших реализацию работы в реальном времени, особенно при использовании ассоциативной памяти.[1]

Структурная схема нейронной сети. Обозначения.

В теории нейронных сетей приняты определённые обозначения. Важно отметить, что эти обозначения действительны и при матричном описании нейронных сетей, например, в *Matlab*. Приняты следующие обозначения:

Обозначения скаляров, векторов и матриц:

скаляры – курсивные строчные буквы: ***a, b, c***;

векторы – прямые строчные полужирные буквы: **a, b, c**;

матрицы – прямые прописные полужирные буквы: **A, B, C**.

Обозначения весовых матриц:

весовая матрица – **W(t)**;

элемент матрицы – $\mathbf{w_{ij}(t)}$, где i – номер строки, j – номер столбца, t – время или итерация;

вектор-столбец – $\mathbf{w_j(t)}$ (вектор, соответствующий столбцу j матрицы W);

вектор-строка – $\mathbf{w_i(t)}$ (вектор, соответствующий строке i матрицы W);

вектор смещений – **b(t)**;

элемент вектора смещений – $\mathbf{b_i(t)}$.

Обозначения для слоев нейронной сети:

верхний индекс из одного символа применяется для того, чтобы указать принадлежность некоторого элемента слою. Например, вектор входа слоя 3 обозначается как $\mathbf{p^3}$;

верхний индекс из двух символов применяется для того, чтобы указать источник сигнала *(l)* и пункт назначения *(k)*; он используется для обозначения матриц весов входа $\mathbf{IW^{k,l}}$ и матриц весов слоя $\mathbf{LW^{k,l}}$. Например, матрица весов от слоя 2 к слою 4 в массиве ячеек всех весов сети будет обозначаться как $\mathbf{LW^{42}}$, и располагаться в ячейке **LW{4,2}**.

Пример архитектуры трёхслойной нейронной сети представлен на рисунке 8.

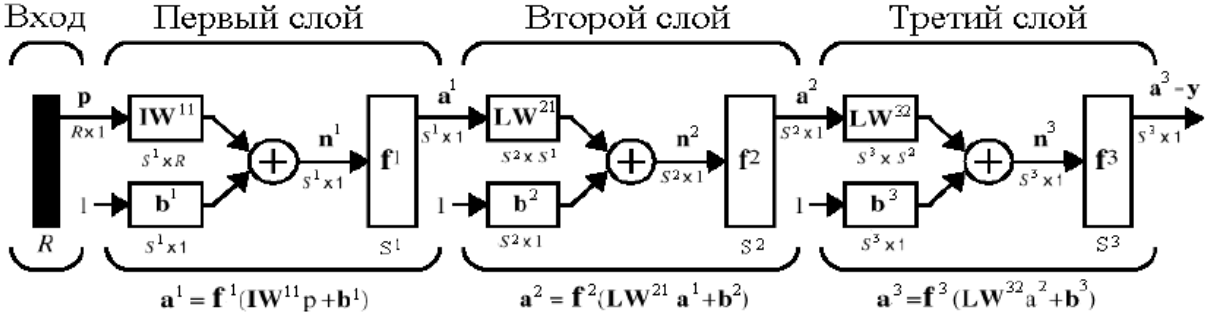


Рисунок 8. Пример архитектуры трёхслойной нейронной сети.

Здесь стоит отметить, что смещение суммируется со взвешенным входом каждого нейрона $\mathbf{n^i}$ и приводит к сдвигу аргумента функции активации на величину **b**. Действие смещения можно свести к схеме, в которой каждый нейрон имеет второй входной сигнал со значением, равным 1. Вход функции активации нейрона по-прежнему остается скалярным и равным сумме взвешенного входа и смещения **b**. Эта сумма является аргументом функции активации **f**, а выходом функции активации является сигнал **a**. Константы **w** и **b** являются скалярными параметрами нейрона. Если в слое количество нейронов больше 1, то значения **w** и **b** хранятся в виде матриц. [2]

Архитектура нейронной сети для управления исполнительным устройством.

В лабораторной работе используется нейронная сеть, структурная схема которой представлена на рисунке 5. На рисунках 9, 10 показаны более подробная структурная схема и архитектура нейронной сети, состоящей в общей сложности из 4 слоёв. Под блоками TDL подразумеваются временные задержки, накладываемые на входные сигналы и сигналы обратных связей.

Сигналы $e_c(t)$ и $e_p(t)$ используются для обучения нейронной сети регулятора и нейронной сети модели объекта управления.

В данном случае для управления исполнительным устройством выбран адаптивный метод управления по эталонной модели (MRAC - model reference adaptive control). В этом случае нет необходимости знать точную математическую модель объекта управления. Также этот метод удобно реализовать в Matlab, в частности в командной строке, так как в Simulink модель такой системы была бы достаточно громоздкой.

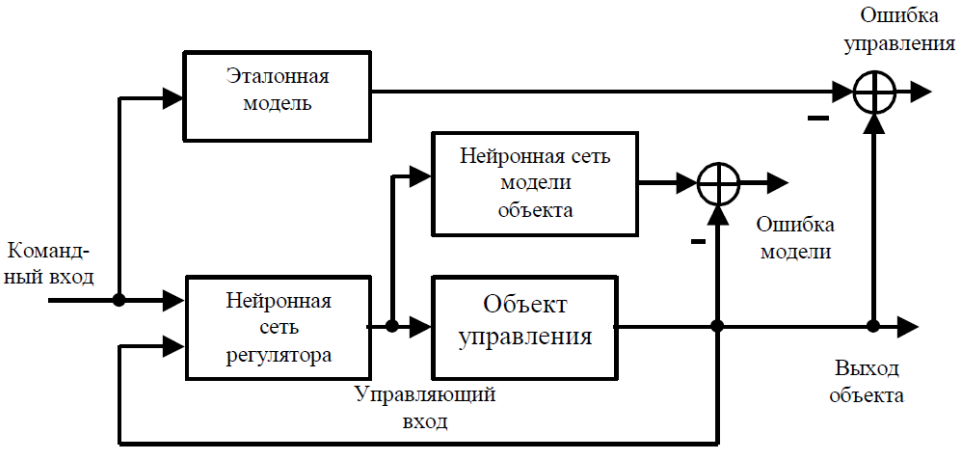


Рисунок 9. Структурная схема системы управления.

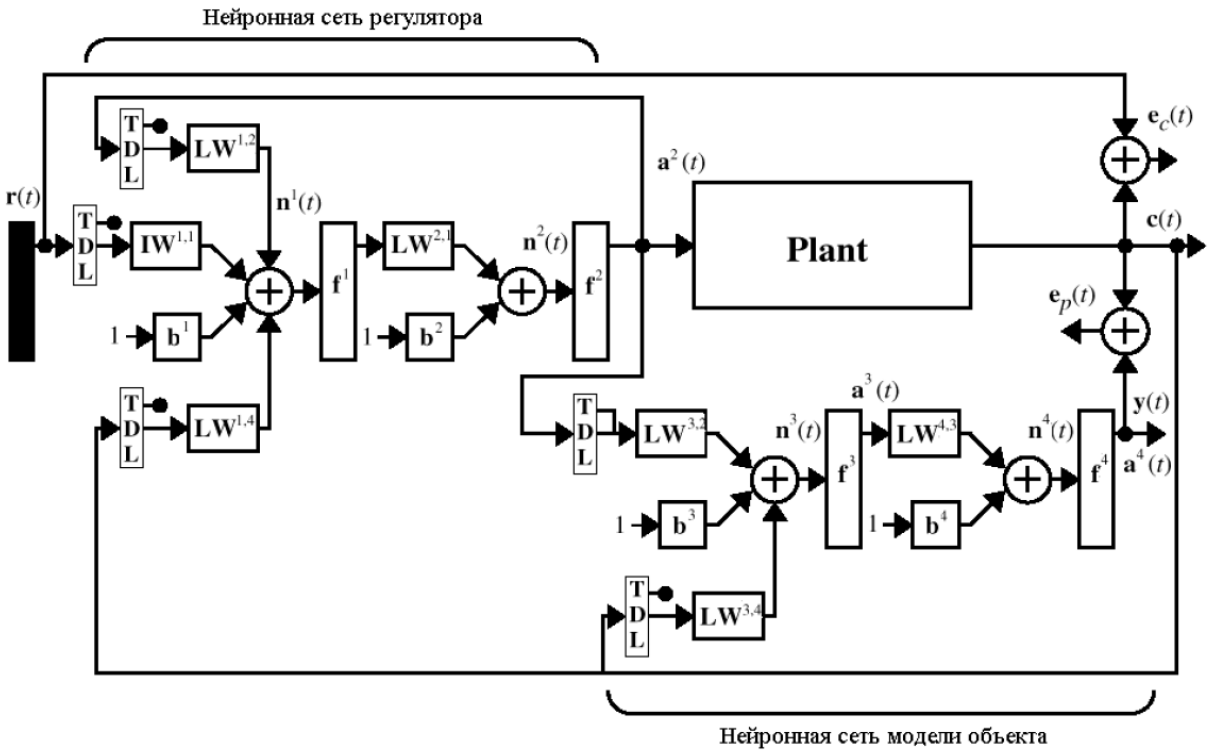


Рисунок 10. Архитектура нейронной сети.

Применение нейронных сетей для решения задач управления позволяет выделить 2 этапа проектирования:

- этап идентификации управляемого процесса;
- этап синтеза закона управления.

В связи с этим архитектура нейронной сети на основе эталонной модели включает две нейронные сети (физически это одна сеть, состоящая из двух подсетей): сеть регулятора, состоящая из 2 слоёв, и сеть модели объекта управления, также состоящая из 2 слоёв, как показано на рисунке 10. Процесс создания такой сети состоит из следующих пунктов:

1) прежде всего, создаётся двухслойная сеть, которая решает задачу идентификации модели объекта управления, на основе нелинейной авторегрессионной модели с внешними входами (NARX - the nonlinear autoregressive network with exogenous inputs). Это периодическая динамическая сеть с обратными связями. Архитектура двухслойной сети NARX представлена на рисунке 11.

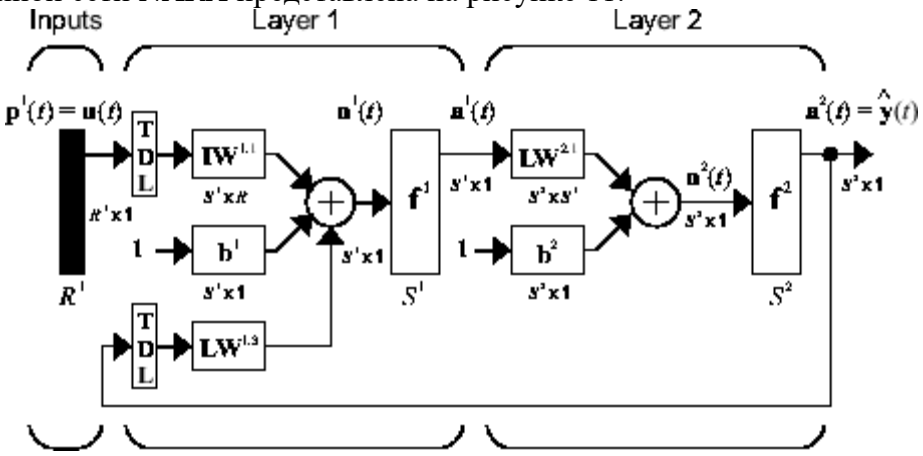


Рисунок 11. Архитектура нейронной сети NARX.

2) при обучении сети на её вход подаются произвольные задающие воздействия, подаваемые также на объект управления. Сигналы, получаемые с выхода объекта управления, подаются на сеть в качестве целевого значения, которое должно получиться на выходе нейронной сети в этот момент времени. После обучения поведение нейронной сети практически полностью повторяет поведение объекта управления.
3) сеть расширяется, к её входам добавляются слои нейронной сети регулятора. Замыкаются обратные связи.
4) при обучении расширенной сети уже настроенные веса подсети модели объекта не изменяются (фиксируются), а на вход подсети регулятора подаются произвольные задающие воздействия, подаваемые также на эталонную модель. Сигналы, получаемые с выхода эталонной модели, подаются на сеть в качестве целевого значения, которое должно получиться на выходе общей нейронной сети в этот момент времени. Таким образом, нейронная подсеть регулятора обучается управлению подсетью модели объекта с целью повторить поведение эталонной модели.
5) После обучения выход подсети регулятора подсоединяется к объекту управления, и сеть проверяется на работоспособность и устойчивость.

Также в *Matlab* в пакете Neural Network Toolbox представлены 3 вида архитектур нейронных сетей, используемых для управления, в виде следующих контроллеров:

1) контроллер с предсказанием (NN Predictive Controller);

2) контроллер на основе модели авторегрессии со скользящим средним (NARMA-L2 Controller);
3) контроллер на основе эталонной модели (Model Reference Controller), используемый в данной работе.

При управлении с предсказанием создаётся нейронная сеть, используемая для предсказания будущего поведения объекта управления, а для расчёта управления применяется алгоритм оптимизации.

При управлении на основе модели авторегрессии со скользящим средним регулятор представляет собой достаточно простую реконструкцию модели объекта управления, полученной на этапе идентификации. Этот регулятор требует наименьшего объёма вычислений, но его недостаток состоит в том, что модель процесса должна быть задана в канонической форме пространства состояния, которой соответствует сопровождающая матрица, что может приводить к вычислительным погрешностям.

При управлении на основе эталонной модели регулятор – это нейронная сеть, которая обучена управлять процессом так, чтобы он отслеживал поведение эталонного процесса. При этом модель управляемого процесса активно используется при настройке параметров самого регулятора. [2]

Алгоритм обучения нейронной сети.

Основной принцип обучения нейронной сети состоит в настройке параметров нейронов таким образом, чтобы поведение сети соответствовало некоторому желаемому поведению. Регулируя веса и параметры смещения, можно обучить сеть выполнять конкретную работу.

Для обучения нейронной сети в данной работе используется алгоритм Левенберга – Марквардта. Данный алгоритм является одним из наиболее широко применяемых алгоритмов для обучения многослойных нейронных сетей. При решении задач аппроксимации функций во многих случаях он является оптимальным между требованиями к вычислительным ресурсам и быстродействием. [3]

“Алгоритм реализует вычисление матрицы Гессе (матрицы вторых частных производных функции) через матрицу Якоби производных от функционала ошибки. В предположении, что функционал определяется как сумма квадратов ошибок, что характерно при обучении нейронных сетей с прямой передачей, гессиан может быть приближенно вычислен как

$$H \cong J^T J,$$

а градиент рассчитан по формуле

$$g = J^T e,$$

где $J = \frac{\partial J}{\partial W}$ – матрица Якоби производных функционала ошибки по настраиваемым параметрам; e – вектор ошибок сети. Матрица Якоби может быть вычислена на основе стандартного метода обратного распространения ошибки. При использовании алгоритма обратного распространения ошибки сеть рассчитывает возникающую в выходном слое ошибку и вычисляет вектор градиента как функцию весов и смещений. Этот вектор указывает направление кратчайшего спуска по поверхности для данной точки, поэтому если продвинуться в этом направлении, то ошибка уменьшится. Последовательность таких шагов в конце концов приведет к минимуму того или иного типа.

Алгоритм Левенберга – Марквардта использует аппроксимацию гессиана следующего вида:

$$x_{k+1} = x_k - (J^T J + \mu I)^{-1} J^T e_k$$

Когда коэффициент μ равен 0, мы получаем метод Ньютона с приближением гессиана в форме $H \cong J^T J$; когда значение μ велико, получаем метод градиентного спуска с маленьким шагом.

Метод Ньютона для ускоренного обучения нейронных сетей определяется соотношением

$$x_{k+1} = x_k - H_k^{-1} g_k,$$

где x_k - вектор настраиваемых параметров; H_k матрица Гессе вторых частных производных функционала ошибки по настраиваемым параметрам; g_k – вектор градиента функционала ошибки.

Поскольку метод Ньютона имеет большую точность и скорость сходимости вблизи минимума, задача состоит в том, чтобы в процессе минимизации как можно быстрее перейти к методу Ньютона. С этой целью параметр μ уменьшают после каждой успешной итерации и увеличивают только тогда, когда пробный шаг показывает, что функционал ошибки возрастает. Такая стратегия обеспечивает уменьшение ошибки после каждой итерации алгоритма.

В *Matlab* в Neural Network Toolbox реализация алгоритма Левенберга – Марквардта в функции trainlm выполняет процедуру обучения, если функции взвешивания, накопления и активации имеют производные. Для вычисления якобиана критерия качества обучения по переменным веса и смещения используется метод обратного распространения ошибки. Каждая настраиваемая переменная корректируется в соответствии с методом Левенберга – Марквардта:

jj = jX * jX;

je = jX * E;

dX = –je/(jj + I * mu),

где E – матрица ошибок; I – единичная матрица.

Параметр адаптации mu возрастает с коэффициентом mu_inc до тех пор, пока изменение весов и смещений dX не приведет к уменьшению критерия качества; после этого коэффициент mu_inc переключается на mu_dec.

Параметр mem_reduc позволяет находить компромисс между объемами оперативной памяти и быстродействием, необходимыми для вычисления якобиана. Когда параметр mem_reduc равен 1, обеспечивается максимальное быстродействие, но и требуются большие объемы памяти. Если увеличить значение mem_reduc вдвое, то потребная память также уменьшится примерно вдвое, но скорость вычислений несколько замедлится. Чем больше значение mem_reduc, тем меньше требования к памяти, но время вычислений существенно увеличивается.

В Matlab функция вызова алгоритма Левенберга – Марквардта для обучения нейронной сети имеет следующие параметры:

Show Training Window Feedback	showWindow: true	Показывать GUI обучения сети
Show Command Line Feedback	showCommandLine: false	Генерировать вывод в командную строку
Command Line Frequency	show: 25	Количество циклов между отображениями
Maximum Epochs	epochs: 1000	Предельное число циклов обучения
Maximum Training Time	time: Inf	Максимальное время обучения, с
Performance Goal	goal: 0	Предельное значение функции качества
Minimum Gradient	min_grad: 1e-07	Минимальное значение градиента критерия качества

Maximum Validation Checks	max_fail: 6	Отношение ошибки контрольного подмножества к ошибке обучающего
Mu	mu: 0.001	Начальное значение mu
Mu Decrease Ratio	mu_dec: 0.1	Коэффициент уменьшения mu
Mu Increase Ratio	mu_inc: 10	Коэффициент увеличения mu
Maximum mu	mu_max: 1e10	Максимальное значение mu

Обучение прекращается, когда выполнено одно из следующих условий:

- значение функции качества стало меньше предельного;
- градиент критерия качества стал меньше значения min_grad;
- достигнуто предельное число циклов обучения;
- превышено максимальное время, отпущенное на обучение;
- ошибка контрольного подмножества превысила ошибку обучающего более чем в max_fail раз.”

[2, стр.84-87, 371-374]

Ход выполнения работы.

1) Запустите *Matlab* и *Simulink*. По уравнению (1) создайте динамическую модель исполнительного устройства, управляемого по моменту привода, считая выходной переменной угол поворота звена. При этом модель должна иметь такую структуру, из которой возможно взять сигнал как угла поворота звена, так и его скорости.
2) Создайте нейронную сеть NARX 4 – 10 – 1 (4 входа, 10 нейронов в скрытом слое, 1 выход). Архитектура сети представлена на рисунке 11 и продублирована на рисунке 12.

   Сеть имеет 1 внешний вход с временной задержкой [0 1] (т. е. вход 1 без задержки и вход 2 с задержкой на один такт) и одну обратную связь с задержкой [1 2] (т. е. вход 3 с задержкой на 1 такт и вход 4 с задержкой на 2 такта).

   Между входами 1,2 и нейронами скрытого слоя имеется матрица весовых коэффициентов IW размерностью 10x2.

   Между входами 3,4 и нейронами скрытого слоя имеется матрица коэффициентов LW1_2 размерностью 10x2.

   Также в скрытом слое имеется матрица смещений B1 размерностью 10x1.

   Функция активации скрытого слоя – гиперболический тангенс, определяемая следующим выражением:

$$\text{tansig(n)} = \frac{2}{1+e^{-2n}} - 1$$

   Между скрытым и входным слоями имеется матрица коэффициентов LW2_1 размерностью 1x10. Также в выходном слое имеется матрица смещений B2 размерностью 1x1. Функция активации выходного слоя – линейная. Также на выходе сети необходимо поставить блок единичной задержки (фиксатор) на один такт.

   Длительность всех задержек примем равной 50 мкс.

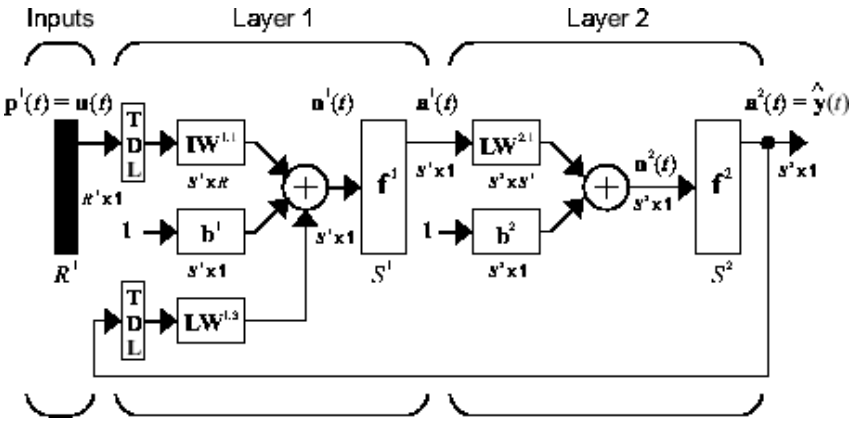


Рисунок 12. Архитектура нейронной сети NARX.

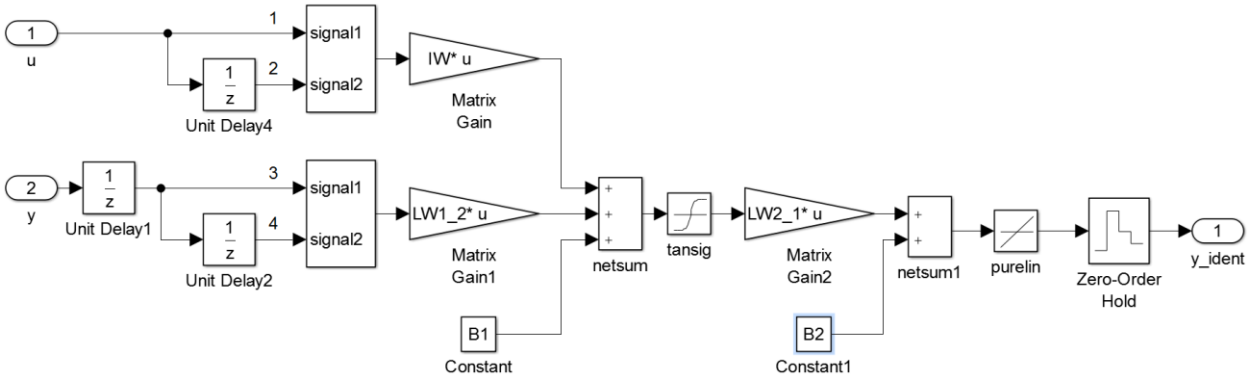


Рисунок 13. Модель нейронной сети для идентификации ОУ.

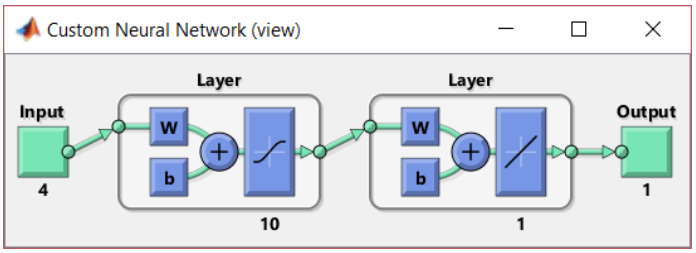


Рисунок 14. Нейронная сеть NARX в *Matlab*.

3) После создания нейронной сети NARX расширьте сеть, добавив модель нейронной сети регулятора 5 – 13 – 1.

Сеть имеет 1 внешний вход с временной задержкой [0 1] и две обратные связи, представленные на рисунках 15 и 16, с задержками [1] и [1 2].

Между входами 1,2 и нейронами скрытого слоя имеется матрица весовых коэффициентов IW_r размерностью 13x2.

Между входом 3 и нейронами скрытого слоя имеется матрица коэффициентов IW_u размерностью 13x1.

Между входами 4,5 и нейронами скрытого слоя имеется матрица весовых коэффициентов IW_y размерностью 13x2.

Также в скрытом слое имеется матрица смещений B1_c размерностью 13x1. Функция активации скрытого слоя – гиперболический тангенс.

Между скрытым и входным слоями имеется матрица коэффициентов LW_c размерностью 1x13. Также в выходном слое имеется матрица смещений B2_c размерностью 1x1. Функция активации выходного слоя – линейная. На выходе сети необходимо ограничить выходной сигнал в пределах ±15 Н*м. И также на выходе сети необходимо поставить блок единичной задержки (фиксатор) на один такт.

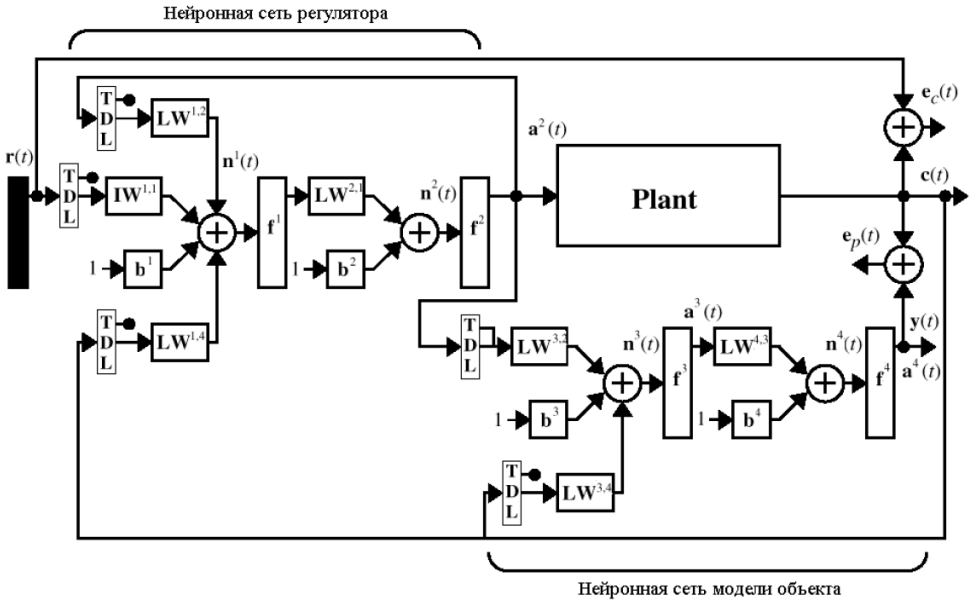


Рисунок 15. Модель нейронной сети системы управления в *Matlab*.

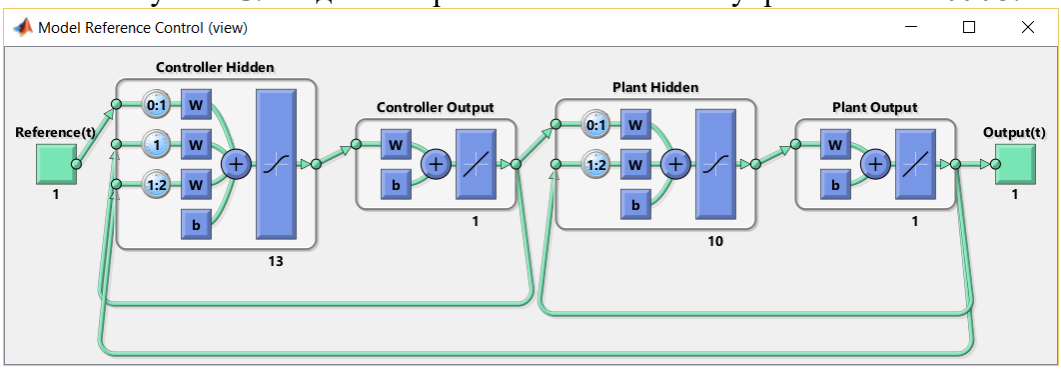


Рисунок 16. Модель нейронной сети системы управления в *Matlab*.

4) Обучение обеих сетей проводится в демонстративном режиме. После обучения занесите значения полученных весовых коэффициентов и смещений в виде вышеперечисленных матриц в рабочее пространство *Matlab*.
5) Соедините нейронные сети в соответствии с рисунками 15 и 16, как показано на рисунке 17.

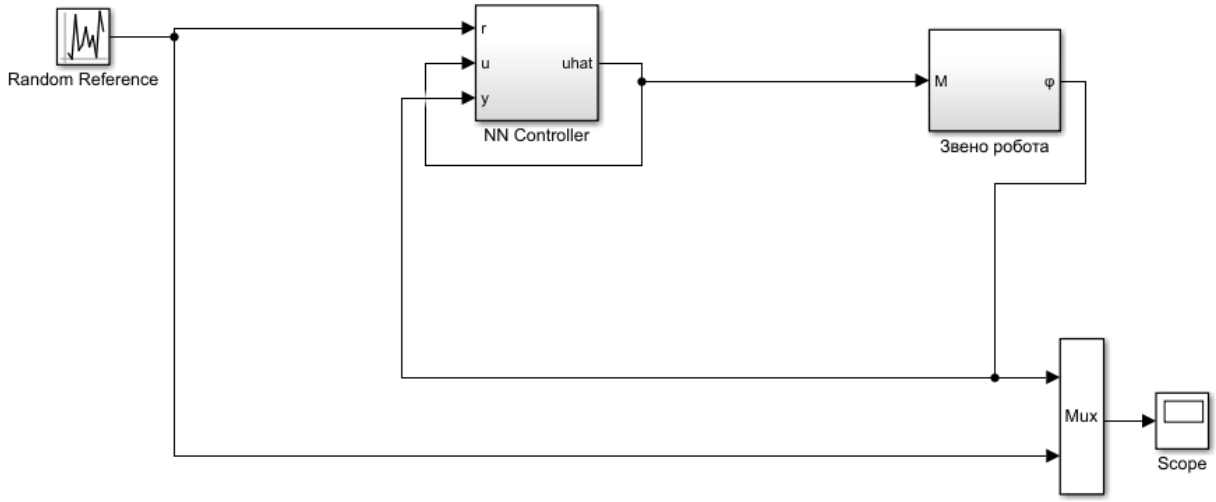


Рисунок 17. Система управления.

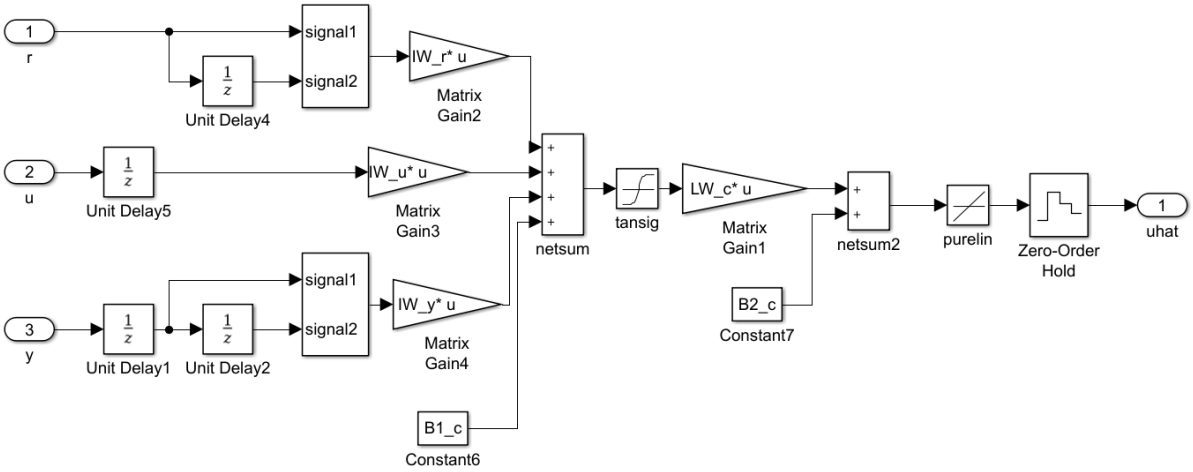


Рисунок 18. Модель нейронной сети системы управления.

В блоках Unit Delay и Zero-Order Hold необходимо проставить нулевые начальные условия и длительность задержек, равную 0.05 с.

6) На вход системы подавайте каждые 10 секунд значения в пределах от -0.5 рад до +0.5 рад. Убедитесь, выполняются ли требования таблицы 2.

Контрольные вопросы.

1) Как можно улучшить качество системы управления?
2) Как установить требования к качеству переходных процессов при нейросетевом управлении?
3) Какие сети можно использовать для управления динамическими объектами?
4) Для чего в нейронной сети необходимы обратные связи?
5) Для чего в нейронной сети необходимы блоки задержки?
6) Зачем используются 2 нейронные сети в данной работе?

Список литературы:

1) Комашинский В. И., Смирнов Д. А., Нейронные сети и их применение в системах управления и связи. – М.: Горячая линия – Телеком, 2003 – 94 с.
2) Медведев В. С., Потемкин В. Г. Нейронные сети. MATLAB 6./Под общ. ред. В. Г. Потемкина. – М.: ДИАЛОГ-МИФИ, 2001. – 630 с.
3) Udemy - Neural Networks made easy with Matlab, MOOC