МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В.Н.Каразіна

Навчально-науковий інститут «Українська інженерго-педагогічна академія»
Кафедра Автоматизації, метрології та енергоефективних технологій

КВАЛІФІКАЦІЙНА РОБОТА

магістра

на тему

СИСТЕМИ УПРАВЛІННЯ ОСНОВНИМИ МЕХАНІЗМАМИ МОСТОВОГО КРАНА З НЕЙРОРЕГУЛЯТОРОМ NN PREDICTIVE CONTROLLER. ТЕМА КОМПЛЕКСНА. СПЕЦІАЛЬНА ЧАСТИНА. СИНТЕЗ І ДОСЛІДЖЕННЯ НЕЙРОМЕРЕЖЕВОЇ СИСТЕМИ УПРАВЛІННЯ МЕХАНІЗМОМ ПЕРЕМІЩЕННЯ МОСТА

(тема кваліфікаційної роботи)

Виконав: студент 2 курсу, групи ДЕА-А24мг
спеціальності: 174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка
(код і найменування спеціальності)

_____/__Олександр ПОПОВ_
(підпис) (ім'я та прізвище)

Керівник _____/_Тетяна ВАСИЛЕЦЬ_
(підпис) (ім'я та прізвище)

Рецензент _____/_Костянтин БРОВКО_
(підпис) (ім'я та прізвище)

«До захисту допущено»

Завідувач кафедри _____/Геннадій КАНЮК
(підпис) (ім'я та прізвище)

Нормоконтроль_____/_Євген КЛЮЧКА__
(підпис) (ім'я та прізвище)

Секретар ЕК _____/__Євген КЛЮЧКА__
(підпис) (ім'я та прізвище)

Харків – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В.Н.Каразіна

Навчально-науковий інститут «Українська інженерго-педагогічна академія ____
Кафедра____Автоматизації, метрології та енергоефективних технологій _____
рівень вищої освіти _____другий (магістрський)_____
Спеціальність ___174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка _____
Освітня програма Автоматизація та комп'ютерно-інтегровані технології _____

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____
(підпис)

д.т.н., проф. Геннадій КАНЮК
(науковий ступінь, вчене звання, ім'я, прізвище)

«___»_____2025р.

ЗАВДАННЯ
на кваліфікаційну роботу магістерського рівня вищої освіти

студенту (ці)_____Олександр ПОПОВ _____
(ім'я, прізвище)

1. Тема роботи _Системи управління основними механізмами мостового крана з нейрорегулятором NN Predictive Controller. Тема комплексна. Спеціальна частина. Синтез і дослідження нейромережевої системи управління механізмом переміщення моста _____
затверджена наказом по університету № 4801-5/3664 від "06" жовтня 2025 р.
2. Термін здачі закінченої роботи "_5_" _грудня__2025 р.
3. Вихідні дані до роботи Вихідні дані до проекту (роботи) Вантажопідйомність крану $G_{вп}=37\cdot10^3$ кг; продуктивність крану Q=55 кг/с; маса моста без візка $m_м=28\cdot10^3$ кг; маса візка $m_в=8\cdot10^3$кг; діаметр коліс моста $D_м$=0,71 м; діаметр коліс візка $D_в$=0,4 м; швидкість пересування моста $V_м$=1,25 м/с; швидкість пересування візка $V_в$=0,65 м/с; швидкість підйому $V_п$=0,1 м/с; маса вантажозахватного пристосування $m_0=1\cdot10^3$кг; середня висота підйому і спуску $L_п$=6 м; середній шлях моста в одну сторону $L_м$=45 м; середній шлях візка в одну сторону $L_в$=24 м; діаметр барабана механізму підйому D=0,65 м. Перерегулювання швидкості по задаючій дії більше 10%, час регулювання до 5 с. _____
4. Зміст роботи (перелік питань, що їх належить розробити). Обґрунтувати перспективність застосування нейромережевих технологій для управління багатомасовою електромеханічною системою механізму пересування моста мостового крана. Розрахувати систему підлеглого регулювання з внутрішнім контуром струму і зовнішнім контуром ЕРС.Розробити математична модель двомасової системи. Виконати моделювання на ЕОМ та провести аналіз показників якості перехідних процесів змінних стану двомасової системи. Розробити структурну схему нейромережевої системи управління. Виконати синтез нейрорегулятора на ЕОМ з застосуванням пакету Neural Network Toolbox системи MATLAB. Провести аналіз результатів моделювання системи з нейрорегулятором. _____
5. Перелік графічного матеріалу (презентаційний матеріал) 1. Магістерська кваліфікаційна робота. Формальні ознаки. 2. Механізм пересування моста. Схема кінематична. 3. . Нейронні мережі. Структурні схеми. 4. Навчання нейронних мереж. Метод зворотного розповсюдження помилки. 5. Нейронні мережі. Ідентифікація об'єктів. 6. Системи нейрорегулювання. Структурні схеми. 7. Система управління. Перехідні процеси. 8. Двомасова система. Математична модель. 9. Двомасова система. Перехідні процеси 10. Регулятор з прогнозом. Структурна схема. 11. Система з нейрорегулятором. Вікна завдання параметрів. 12. Нейромережева модель об'єкту. Структурна схема. 13. Нейромережева система. Перехідні процеси. _14,15. Магістерська кваліфікаційна робота. Висновки.. _____

6. Консультант:

Розділ	Консультант	Підпис, дата		Оцінка (бали)
		Завдання видав	Завдання прийняв	

7. Дата видачі завдання “ 9 вересня 2025 р.

Керівник _____ __Тетяна ВАСИЛЕЦЬ_
(підпис) (ім’я, прізвище)

Завдання прийняв до виконання _____ __Олександр ПОПОВ__
(підпис) (ім’я, прізвище)

КАЛЕНДАРНИЙ ПЛАН-ГРАФІК
виконання кваліфікаційної роботи

№ з/п	Назва етапів роботи та питань, які мають бути розроблені відповідно до завдання	Термін виконання	Позначки керівника про виконання завдань
1	Обґрунтування застосування нейромережевих технологій для управління багатомасовою електромеханічною системою механізму переміщення моста мостового крана.	09.09.2025-18.09.2025	
2	Розрахунок системи підлеглого регулювання координат електроприводу з оптимізацією контурів за модульним ти симетричним критеріями. Моделювання системи на ЕОМ.	19.09.2025-15.10.20254	
3	Дослідження системи з урахуванням пружних елементів. Розробка математичної моделі двомасової системи та моделювання двомасової системи на ЕОМ.	16.10.2025-05.11.2025	
4	Синтез нейромережевої системи управління на ЕОМ з застосуванням пакету Neural Network Toolbox системи MATLAB. Моделювання нейромережевої системи.	06.11.2025-25.11.2025	
5	Оформлення роботи	26.11.2025-05.12.2025	

Студент (ка) _____ ___Олександр ПОПОВ___
(підпис) (ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка складається з: 215 сторінки, 68 рисунків, 4 таблиці, 102 використаних джерел, 4 додатки.

Метою дослідження є створення та обґрунтування методів синтезу системи керування електроприводом механізму пересування моста мостового крана, спрямованих на підвищення показників якості роботи системи шляхом застосування нейронорегуляторів.

У роботі аналітичним шляхом, із залученням базових принципів теорії автоматичного керування, визначено параметри структурних елементів системи. Методи математичного моделювання застосовано для побудови моделей функціонування багатомасових електромеханічних систем. На основі інструментів теорії штучних нейронних мереж виконано синтез нейрорегулятора для керування двомасовою електромеханічною моделлю. За допомогою комп'ютерного моделювання в середовищі MATLAB досліджено динамічні властивості та якість функціонування нейромережевої системи.

У результаті проведеного магістерського дослідження науково доведено ефективність та доцільність впровадження нейромережевих підходів у системи керування багатомасовими електромеханічними об'єктами, зокрема механізмом переміщення моста мостового крана. Розроблено систему керування координатами електроприводу, здійснено розрахунок параметрів автоматичної системи керування та проведено моделювання її роботи на ЕОМ.

Створено математичну модель електромеханічної системи з урахуванням пружних зв'язків у механічній частині приводу. Визначено динамічні характеристики двомасової системи. Проаналізовано архітектуру нейромережевої системи керування та виконано синтез нейронного регулятора типу NN Predictive Controller у середовищі MATLAB. Проведено моделювання роботи нейромережевої системи керування з використанням розробленого регулятора, що дозволило встановити високі якісні показники її функціонування.

Основні результати магістерської роботи впроваджено в освітній процес кафедри АМЕТ ННІ «УІПА» ХНУ імені В. Н. Каразіна та використовуються під час читання лекцій, виконання курсових проєктів і проведення практичних занять.

Ключові слова: ШТУЧНИЙ НЕРОН, НЕЙРОННІ МЕРЕЖІ, МОСТОВИЙ КРАН, ЕЛЕКТРОПРИВОД МЕХАНІЗМУ ПЕРЕМІЩЕННЯ МОСТА, СИСТЕМА ПІДЛЕГЛОГО УПРАВЛІННЯ, ОДНОМАСОВА І ДВОМАСОВА СИСТЕМИ, НЕЙРОМЕРЕЖЕВА СИСТЕМА, НЕЙРОРЕГУЛЯТОР

ABSTRACT

An explanatory message consists of: 215 pages, 68 figures, 4 tables, 102 used sources, и 4 additions.

The purpose of the research is to create and substantiate methods for synthesizing the control system of the electric drive of the bridge movement mechanism of the bridge crane, aimed at improving the quality of the system's operation by using neural regulators.

In the work, the parameters of the structural elements of the system were determined analytically, using the basic principles of the theory of automatic control. Mathematical modeling methods were used to build models of the functioning of multi-mass electromechanical systems. Based on the tools of the theory of artificial neural networks, a neural regulator was synthesized to control a two-mass electromechanical model. Using computer modeling in the MATLAB environment, the dynamic properties and quality of functioning of the neural network system were investigated.

As a result of the conducted master's research, the effectiveness and feasibility of implementing neural network approaches in the control system of multi-mass electromechanical objects, in particular the mechanism of moving the bridge of the bridge crane, were scientifically proven. A coordinate control system for the electric drive was developed, the parameters of the automatic control system were calculated, and its operation was simulated on a computer.

A mathematical model of an electromechanical system was created taking into account elastic connections in the mechanical part of the drive. The dynamic characteristics of a two-mass system were determined. The architecture of the neural network control system was analyzed and a neural regulator of the NN Predictive Controller type was synthesized in the MATLAB environment. The operation of the neural network control system was simulated using the developed regulator, which allowed establishing high quality indicators of its functioning.

The main results of the master's thesis were implemented in the educational process of the Department of AMET of the NNI "UIPA" of the V. N. Karazin KhNU and are used during lectures, course projects and practical classes.

Keywords: ARTIFICIAL NERO, NEURAL NETWORKS, BRIDGE CRANE, ELECTRIC DRIVE OF THE MOVING BRIDGE MECHANISM, SYSTEM OF SLAVE CONTROL, ONE-MASS AND TWO-MASS SYSTEM, NEURAL NETWORK SYSTEM, NEUROREGULATOR

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

$P_{н}$ – номінальна потужність;

$U_{н}$ – номінальна напруга;

$I_{н}$ – номінальний струм;

$M_{н}$ – номінальний момент;

$n_{н}$ – номінальна швидкість обертання;

$\Phi_{н}$ – номінальний магнітний потік;

R – активний опір;

L – індуктивність;

J – момент інерції;

$k_{д}$ – коефіцієнт посилення двигуна;

$T_{е}$ – електромагнітна постійна часу електроприводу;

$T_{м}$ – електромеханічна постійна часу електроприводу;

$W(p)$ – передатна функція;

c – коефіцієнт жорсткості;

β – коефіцієнт внутрішнього в'язкого тертя;

A – матриця стану;

B – матриця управління;

C – матриця виходу;

$\vec{X}(t)$ – вектор стану системи;

$\vec{U}(t)$ – вектор управління;

$\vec{Y}(t)$ – вектор виходу;

САУ – система автоматичного управління;

М – двигун.

ЗМІСТ

ВСТУП

Актуальність теми. Підвищення рівня автоматизації промислових технологічних комплексів та удосконалення систем керування, що в них застосовуються, ґрунтуються на сучасних науково-технічних досягненнях і відіграють ключову роль у зростанні продуктивності виробництва, підвищенні ефективності використання трудових ресурсів та у забезпеченні стабільної якості продукції. Розвиток автоматизованих систем охоплює впровадження адаптивних методів керування, покращення технологій векторного керування електроприводами змінного струму, активне розширення використання обчислювальних засобів різного рівня складності, розробку схем прямого цифрового управління та інших сучасних підходів. Одним із найбільш перспективних напрямів вважається застосування нейромережевих технологій у системах керування.

Системи управління, в яких використовуються штучні нейронні мережі, належать до класу нелінійних динамічних систем. Нейромережеві моделі здатні виконувати різні функції, зокрема функції адаптивного регулятора для багатозв'язаних нелінійних об'єктів. Можливі два базові принципи їх роботи. У першому випадку нейронна мережа проходить навчання одночасно з формуванням керуючого впливу на виконавчу частину системи, тобто процеси навчання та управління є інтегрованими та орієнтовані на спільний цільовий критерій.

Другий підхід передбачає розділення етапів навчання і керування: спочатку мережа апріорі навчається певному оптимальному закону регулювання в умовах, що відповідають майбутнім режимам роботи, а вже після цього відтворює наближену оптимальну функцію у процесі експлуатації. Цей спосіб, який часто називають супервізорним керуванням, є наразі найбільш уживаним, однак пов'язаний із тим, що синтез і налаштування регулятора відбуваються у режимі off-line.

Необхідність застосування тієї чи іншої стратегії (on-line або off-line) визначається специфікою задачі та безпосередньо впливає на вибір алгоритму на-

вчання — від безпошукових до алгоритмів глобальної оптимізації. У випадках, коли доступні великі масиви даних про поведінку технологічного об’єкта, ефективним стає off-line навчання з використанням генетичних алгоритмів, випадкових пошукових методів або статистичних підходів. Якщо ж властивості об’єкта змінюються під час роботи, доцільніше застосовувати on-line процедури адаптивного налаштування мережі.

Нейронні мережі також можуть використовуватись для ідентифікації станів нелінійних систем, слугуючи основою для створення розширених фільтрів Калмана, або як оптимізатори параметрів традиційних регуляторів. Сукупність моделей і структур, що реалізують такі функції, отримала назву нейромережевих систем керування. Найбільш широке застосування в управлінні технічними об'єктами отримали багатошарові нелінійні мережі [1–12].

Підвищення рівня технічної досконалості електроприводів на основі сучасних методів керування є важливою умовою зростання ефективності виробництва, поліпшення його техніко-економічних показників і забезпечення високої якості випуску промислової продукції.

У вітчизняній промисловості активно розробляються нові типи підйомно-транспортного обладнання, автоматизованих механізмів циклічної дії, конвеєрних систем, що сприяє розширенню ринку технічних засобів для механізації складських та вантажно-розвантажувальних операцій. Кранове обладнання є невід’ємною складовою комплексної механізації виробничих процесів у багатьох галузях.

Більшість сучасних вантажопідйомних машин оснащуються електричними приводами, які забезпечують привід головних механізмів агрегатів. Від їх надійності та технічного рівня залежить ефективність експлуатації всього кранового комплексу. Такі електроприводи працюють у режимах короткочасних навантажень, здійснюють часті пуски, працюють в умовах значних інерційних та динамічних впливів під час розгону і гальмування.

Попри наявність значного досвіду проєктування, експлуатації та модернізації електроприводів мостових кранів, комплексне підвищення ефективності їх

функціонування неможливе без вдосконалення систем керування. Саме система управління визначає здатність електроприводу забезпечувати оптимальні режими руху, досягати високої продуктивності та забезпечувати безвідмовну роботу обладнання. Тому розроблення новітніх систем керування електроприводами мостових кранів є актуальним науково-технічним завданням.

Мета і завдання дослідження

Метою роботи є створення методів синтезу системи керування електроприводом механізму пересування моста мостового крана з метою забезпечення високих показників якості функціонування на основі нейромережевих регуляторів.

Для досягнення поставленої мети необхідно розв’язати такі основні задачі:

1. Провести аналіз сучасних систем управління та визначити можливості впровадження нейромережевих технологій.

2. Розвинути математичну модель динамічних процесів об’єкта управління.

3. Сформувати структурну схему системи управління із включенням нейромережевого регулятора.

4. Виконати синтез нейромережевої системи з прогнозуючим регулятором, здатної забезпечити високий рівень якості керування.

5. Провести моделювання роботи нейромережевої системи на ЕОМ та здійснити оцінку її характеристик.

Об’єкт і предмет дослідження

Об’єктом дослідження є динаміка процесів у системі керування електроприводом механізму пересування моста мостового крана з урахуванням пружних зв’язків.

Предметом дослідження виступають математичні моделі та методи синтезу нейромережевої системи керування двомасовою електромеханічною системою.

Методи дослідження

Теоретична частина роботи базується на фундаментальних положеннях теорії автоматичного управління. Для формування математичних моделей використано методи математичного моделювання. Засоби теорії штучних нейронних мереж дали змогу реалізувати нейромережеву модель та визначити алгоритми її навчання. Методи нейроуправління стали основою для синтезу керуючої системи, а імітаційне моделювання дозволило комплексно оцінити її динамічні властивості та підтвердити ефективність запропонованих рішень.

Наукова новизна отриманих результатів

Наукова новизна полягає у такому:

1. Уперше надано наукове обґрунтування ефективності застосування нейромережевих підходів для управління багатомасовою електромеханічною системою механізму переміщення моста мостового крана.

2. Запропоновано та вперше синтезовано нейромережеву систему управління електроприводом механізму пересування моста з урахуванням пружних елементів, що забезпечує високий рівень якості регулювання.

3. Уперше подано математичну модель системи керування електроприводом у формі багатошарового персептрона, яка враховує пружні властивості механічних ланок.

4. Розвинуто математичні моделі об’єктів управління з комплексними кінематичними взаємозв’язками у вигляді двомасових електромеханічних систем.

5. Отримано подальший розвиток методів синтезу нейромережевих структур із прогнозуванням для керування складними динамічними об’єктами.

Практичне значення

Практична цінність роботи полягає у створенні нейромережевої системи керування двомасовою електромеханічною системою механізму пересування моста мостового крана, що забезпечує якісне й ефективне регулювання технологічних процесів.

Основні результати впроваджено у навчальний процес кафедри АМЕТ ННІ «УІПА» ХНУ ім. В. Н. Каразіна під час викладання лекційних курсів, виконання курсових та проведення практичних занять.

Особистий внесок здобувача

Усі положення, винесені на захист, отримані автором особисто. Серед них: синтез системи регулювання координат електроприводу; розроблення моделі з урахуванням пружних елементів; формування структурної схеми нейромережевої системи; створення нейрорегулятора NN Predictive Controller у середовищі MATLAB (Neural Network Toolbox); виконання моделювання та аналіз результатів.

У роботі наведено коректні посилання на авторів та джерела, використані при запозиченні формул, закономірностей та експериментальних даних.

Апробація результатів

Основні наукові результати дослідження були представлені на студентській науковій конференції ННІ «УІПА» ХНУ ім. В. Н. Каразіна (Харків, 2024 р.).

Структура і обсяг роботи

Робота складається зі вступу, чотирьох розділів, висновків, додатків та списку літературних джерел. Повний обсяг становить 215 сторінки, включаючи 68 рисунків (39 у тексті та 29 на 25 окремих сторінках), 2 таблиці, 4 додатків на 39 сторінках та список літератури з 102 найменувань на 10 сторінках.

РОЗДІЛ 1
ОГЛЯД ЛІТЕРАТУРИ ЗА ТЕМОЮ РОБОТИ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Основні етапи розвитку теорії штучних нейронних мереж

Історія формування штучних нейронних мереж (ШНМ) як самостійного наукового напряму бере початок у середині XX століття. Перші формальні моделі нейронних елементів були розроблені В. Мак-Каллоком і У. Піттсом [13], які у 1943 році запропонували логічну модель нейрона, здатну виконувати базові операції булевої алгебри. Пізніше Ф. Розенблатт [14] представив перцептрон — першу навчальну штучну нейронну мережу, яка стала поштовхом для хвилі оптимізму в кібернетиці та ранній теорії машинного навчання. Попри обмеження перцептрона, які довів Мінський у 1969 році, ці роботи заклали фундамент сучасних нейромережевих технологій.

Уже на початку 1970-х років з'явилися перші спроби застосувати штучні нейронні мережі для керування складними динамічними об'єктами в так званих «біонічних системах» [15, 16]. Дослідники намагалися моделювати принципи роботи біологічних регуляторів, припускаючи, що в нервових структурах живих організмів процеси керування відбуваються в *n*-мірному фазовому просторі, поділеному на області за критерієм виду

$$Q = \min\{\max F(\Psi_i(t), \Psi_i^{(1)}(t), \Psi_i^{(2)}(t), ..., \Psi_i^{(m)}(t))\}, \quad i = \overline{1,n}, \qquad (1.1)$$

де $\Psi_i(t) = x_i(t) - g_i(t)$ – динамічні помилки регулювання;

$\Psi_i^{(i)}(t)$ – i-а похідна від помилки;

$x_i(t)$ – поточне значення регульованої величини;

$g_i(t)$ – її задане значення;

$F(\cdot)$ – функціонал, що визначає вимоги до якості керування.

У такій постановці нейронна регулююча система виконує функцію класифікатора фазового простору, визначаючи, в яку з областей потрапляє поточний стан системи. Перехід у певну область генерує відповідні коригуючі сигнали $\xi_i \Psi_i^{(i)}(t)$, де ξ_i – змінні залежно від положення зображуючої точки усередині цієї області параметри. Таким чином, задача керування зводиться до коректної зміни вектора зворотних зв'язків відповідно до траєкторії у фазовому просторі.

Автори цих концепцій підкреслювали їх «гіпотетичний» характер, однак припускали високу ймовірність того, що саме так організовано керування в біологічних системах. Зокрема, окодвигунна система тварин наводилася як приклад природного нейронного класифікатора, здатного адаптивно регулювати рух у складному багатовимірному просторі.

Попри цікаві теоретичні підходи, на той час відсутні були ефективні методи навчання багатошарових мереж, що стримувало практичне застосування цих ідей.

Етап становлення алгоритмів навчання.

Термін «нейроуправління» вперше запропонував П. Вербос у 1974 році [17], коли описав підхід до обчислення градієнтів у складних структурах керування. Проте реальний прорив відбувся лише наприкінці 1980-х років, коли Д. Румельхард та його колеги [18, 19] фактично перевідкрили і популяризували алгоритм зворотного розповсюдження помилки (backpropagation).

Саме цей алгоритм відкрив можливість навчання багатошарових нейромереж із великою кількістю параметрів, що стало основою всіх сучасних нейронних технологій. У цей час вагомий внесок у розвиток нейромережевого керування зробили Нарендра та його співробітники [20, 21], які описали принципи нейронної ідентифікації та нейронно-адаптивного керування для нелінійних систем.

Розвиток архітектур у 1990-х роках

Починаючи з 1990-х років, теорія штучних нейронних мереж (ШНМ) увійшла у фазу інтенсивного розвитку, який суттєво вплинув на формування сучасних інтелектуальних систем керування та обробки інформації. У цей пері-

од відбувся перехід від відносно простих моделей типу перцептронів до багатошарових мереж, здатних навчатися за допомогою алгоритму зворотного розповсюдження помилки (backpropagation). Застосування цього алгоритму надало можливість ефективно тренувати моделі з великою кількістю параметрів та значною обчислювальною складністю. Саме фундаментальні дослідження Румельхарта, Гінтона та Вільямса [22] (Rumelhart, Hinton & Williams, 1986) стали основою для практичного впровадження методу градієнтного спуску в навчання багатошарових перцептронів, заклавши підґрунтя для подальших наукових робіт і технологічних рішень у 1990-х роках.

У цей же час активно розвиваються нові архітектури нейронних мереж, орієнтовані на різноманітні типи даних і класів задач. До них належать радіально-базисні мережі (RBF), самоорганізувальні карти Кохонена (SOM), рекурентні нейронні мережі (RNN), а також мережі Хопфілда, призначені для розв’язання оптимізаційних задач ([23, 24] Kohonen, 1990; Hopfield, 1991). Поява та активне вивчення цих архітектур стимулювали поглиблення теоретичних досліджень, зокрема у сферах аналізу стійкості, збіжності навчання та здатності нейромереж до узагальнення.

Глибоке навчання 2000–2010-х років.

Починаючи з 2000-х років, основну увагу приділено глибоким нейронним мережам (Deep Neural Networks, DNN), що складаються з великої кількості прихованих шарів. Прорив у цій галузі пов’язують з роботами Лекуна, Хінтона та Бенджіо ([25].LeCun, Bengio & Hinton, 2015), де показано ефективність глибоких згорткових мереж (CNN) для розпізнавання зображень і рекурентних мереж (LSTM, GRU) для обробки часових рядів. Водночас зросла роль обчислювальної потужності (GPU) і великих наборів даних, що зробило можливим практичне навчання таких моделей.

Сучасний етап розвитку (2010–2020-ті роки).

У 2010–2020-х роках теорія ШНМ активно розвивається в напрямку обґрунтування властивостей узагальнення, стабільності, інтерпретованості та енергетичної ефективності**.** З’являються роботи з нейронно-символьних підходів**,**

нейромережевих керуючих систем та глибокого підкріплювального навчання (Deep Reinforcement Learning), що поєднує нейронні мережі з класичною теорією керування ([26, 27].Goodfellow, Bengio & Courville, 2016; Silver et al., 2017).

Таким чином, за три десятиліття - від 1990-х до 2020-х років - теорія штучних нейронних мереж еволюціонувала від відносно простих моделей до потужних гібридних систем, що поєднують машинне навчання, оптимізацію та методи керування складними нелінійними динамічними об'єктами.

Нейрокомп'ютери та міждисциплінарний розвиток

У новітній період стрімкий розвиток субмікронних і нанотехнологій, молекулярної електроніки, біомолекулярних сенсорів створює умови для появи апаратних нейрокомп'ютерів, які імітують принципи функціонування біологічної нервової системи на фізичному рівні.

Дослідження штучних нейронних мереж вимагає синергії різних дисциплін: нейрофізіології, когнітивних наук, психології, статистичної фізики, теорії обчислень, кібернетики, штучного інтелекту, паралельних обчислень та електроніки. Водночас самі ШНМ стають інструментом, який поглиблює та розширює ці дисципліни, сприяючи виникненню нових теорій і технологій.

Практичне застосування нейромережевих систем керування.

На сучасному етапі існує велика кількість реально функціонуючих систем керування на основі нейронних мереж. У науковій літературі описані успішні приклади застосування нейропідходів для керування: літаком [28-29], вертольотом [430], гірничозбагачувальним процесом [31], автомобілем-роботом [32], гібридним двигуном автомобіля [33], пневмоциліндром [34], об'єктом спеціального призначення [35], моделлю перевернутого маятника [36], електропіччю [37], турбогенератором [38], зварювальним апаратом [39], застосування нейронних мереж для теплового комфорту та енергозбереження у громадських будівлях [40], керування системами кондиціонування повітря [41], керування електроприводами [42], керування роботизованими маніпуляторами [43-47] і інших. Ці результати підтверджують, що нейронні мережі є ефективним інструментом для керування складними нелінійними динамічними об'єктами на-

віть у умовах значної невизначеності та зміни структури середовища.

У додатках А–В подано ключові положення теорії штучних нейронних мереж, які слугують методологічною основою для проведених у роботі досліджень.

Для синтезу нейромережевої системи управління в даній роботі застосовано багатошарові прямонаправлені нейронні мережі, у яких передавання сигналів між нейронами відбувається лише у прямому напрямку, без внутрішніх зворотних зв’язків. Така структура відповідає класичному багатошаровому персептрону.

У додатку А висвітлено принципи побудови штучного нейрона, описано конфігурації багатошарових мереж та наведено їх основні властивості.

У додатку Б наведено огляд сучасних підходів до навчання нейромереж і обґрунтовано вибір методу тренування, який використано під час розроблення нейромережевої частини системи керування.

Важливою складовою задачі управління, прогнозування динаміки та моделювання нелінійних об’єктів є побудова коректних математичних моделей. Особливості ідентифікації динамічних систем за допомогою нейромережевих методів детально розглянуто в додатку В.

1.2 Огляд структур систем нейрорегулювання. Вибір структури системи

На сьогоднішній день запропоновано та експериментально випробувано дуже велику кількість різноманітних структур систем нейроуправління. Частина з них ґрунтується на використанні аналітичного закону регулювання в поєднанні з нейромоделлю об’єкта. У таких системах нейронна мережа виконує відновлення окремих елементів матриць математичної моделі, що дозволяє віднести їх до класу структурованих методів керування. Проте максимально повно використати апроксимаційні можливості штучних нейронних мереж здатні неструктуровані методи, які не потребують аналітичного опису об’єкта. Саме тому в подальшому розглядаються найбільш актуальні на сьогоднішній день структури нейромережевих систем керування.

1.2.1 Послідовна схема нейроуправління

Однією з класичних конфігурацій є послідовна (інверсно-пряма) схема нейроуправління. У цьому підході нейронна мережа навчається реалізовувати відображення, яке є оберненим до динаміки об'єкта. Тому таку структуру часто називають інверсним нейрокеруванням. Основним завданням навчання мережі є мінімізація певного функціонала помилки, при цьому в процесі навчання використовується Якобіан об'єкта. Якщо значення помилки стає близьким до нуля, вважається, що нейромережа достатньо точно відтворює інверсну динаміку об'єкта. Такий підхід отримав назву спеціалізованого нейроуправління зі зворотним відображенням, його структурна схема наведена на рис. 1.1.

Приклад побудови нейрорегулятора.

Розглянемо одновимірний динамічний об'єкт керування, для якого відомо співвідношення «вхід–вихід». Як нейрорегулятор використаємо двошаровий персептрон з одним вихідним нейроном. Нехай у прихованому шарі застосовуються логістичні функції активації:

$$f_{log}(u) = \frac{1}{1+e^{-u}}, \tag{1.1}$$

а їх похідні визначаються співвідношенням

$$f'_{log}(u) = f_{log}(1 - f_{log}(u)). \tag{1.2}$$

Структурна схема такого регулятора зображена на рис. 1.2.

Метою є мінімізація квадратичного критерію помилки:

$$E = \frac{1}{2}e^2 = \frac{1}{2}(d(k) - y(k))^2 \tag{1.3}$$

тут $y(k) = f_0(u(k))$, де – f_0 невідома функція перетворення.

Вихідний сигнал нейронної мережі визначається як:

$$u = y_1^2 = f_{log}(a_1^2), \tag{1.4}$$

$$a_1^2 = \sum_{j=1}^{N^1} w_{1j}^2 y_j^1 + b_{10}^2 n_0^1, \tag{1.5}$$

$$y_j^1 = f_{log}(a_j^1), \tag{1.6}$$

$$a_j^1 = w_{j1}^1 d + b_{j0}^1 n_0^0. \tag{1.7}$$

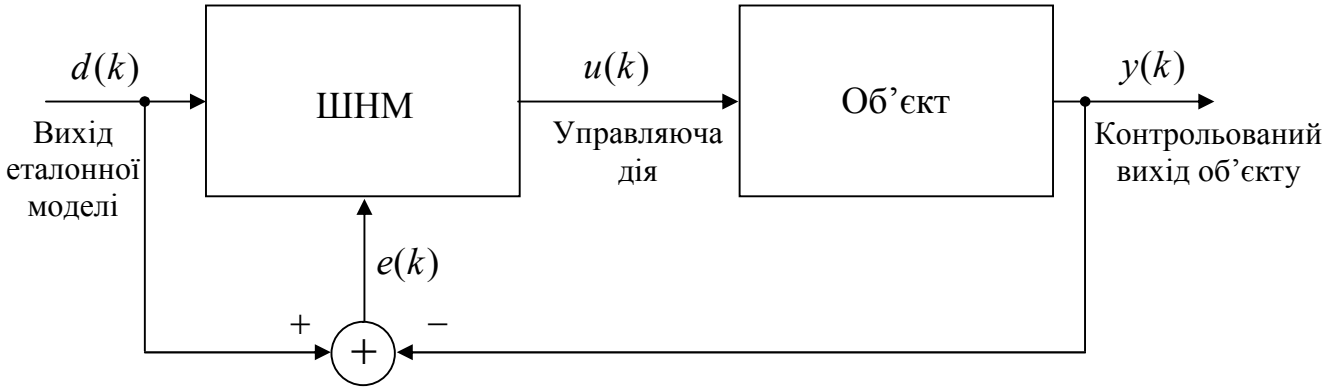


Рисунок 1.1 – Схема реалізації спеціалізованого нейронного керування

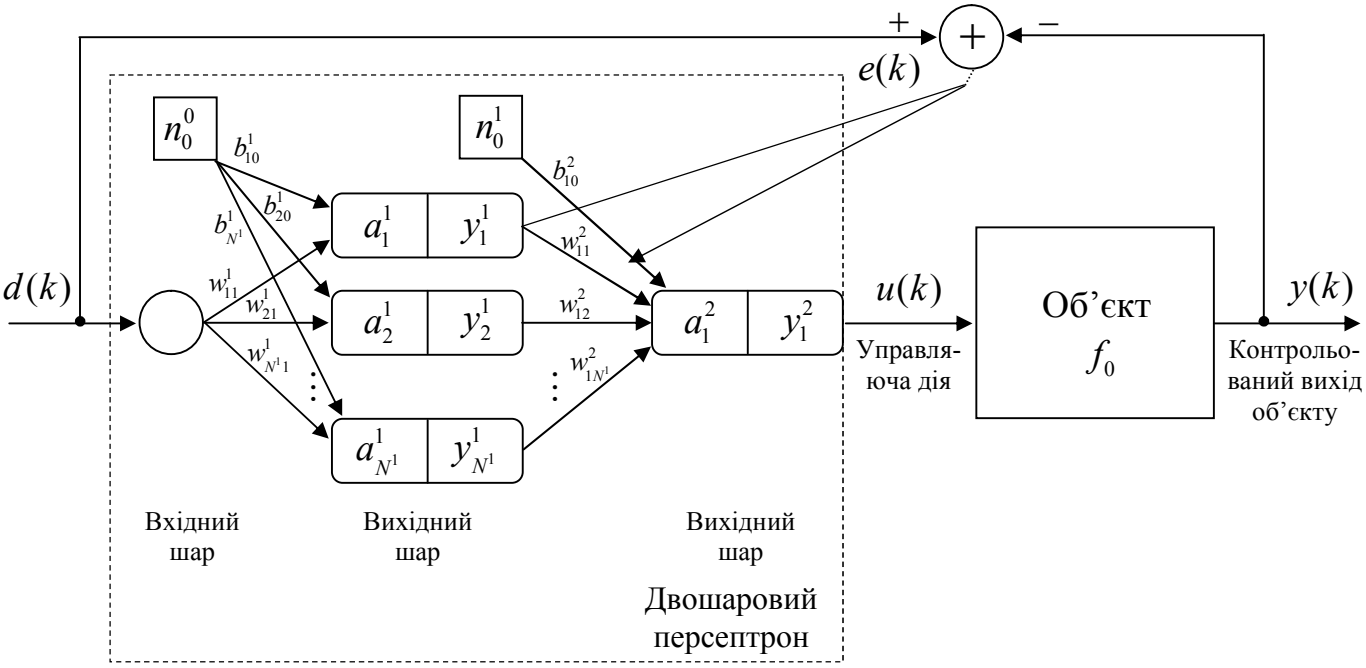


Рисунок 1.2 – Принципова структура нейронного регулятора, побудованого на двошаровому перцептроні

Корекція вагів нейронної мережі.

Для налаштування параметрів використовується метод найшвидшого спуску. З урахуванням правила похідної складної функції формула оновлення ваг вихідного шару набуває вигляду:

$$\Delta w_{1j}^2 = -\eta \frac{\partial E}{\partial w_{1j}^2} = -\eta \frac{\partial E}{\partial a_1^2} \frac{\partial a_1^2}{\partial w_{1j}^2}.$$

З виразу (1.5) випливає:

$$\frac{\partial a_1^2}{\partial w_{1j}^2} = y_j^1.$$

Враховуючи (1.3), (1.1), (1.2), вводимо локальну помилку вихідного шару $\delta^2 = \frac{\partial E}{\partial a_1^2}$:

$$\delta^2 = \frac{\partial E}{\partial e} \frac{\partial e}{\partial y} \frac{\partial y}{\partial u} \frac{\partial u}{\partial a_1^2} = e \frac{\partial y_0}{\partial u} \frac{\partial u}{\partial a_1^2} = e f_0'(u) u(1-u), \tag{1.8}$$

і отримуємо загальну формулу налаштування вагів:

$$\Delta w_{1j}^2 = -\eta \delta^2 y_j^1. \tag{1.9}$$

Для прихованого шару аналогічно:

$$\Delta w_{j1}^1 = -\eta \delta_j^1 d, \tag{1.10}$$

де локальна помилка прихованого шару:

$$\delta_j^1 = \frac{\partial E}{\partial a_1^2} \frac{\partial a_1^2}{\partial y_j^1} \frac{\partial y_j^1}{\partial a_j^1} = \delta^2 w_{1j}^2 y_j^1 (1 - y_j^1). \tag{1.11}$$

Алгоритм навчання нейрорегулятора.

1. Ініціалізація: задаємо початкові значення вагів w, зсувів b та норму навчання η.

2. Обчислення помилки: визначаються δ^2 та δ_j^1 за (1.8) і (1.11), використовуючи вихід еталонної моделі $d(k)$, значення керувального сигналу $u(k)$ та вихід об'єкта $y(k)$.

3. Оновлення вагів вихідного шару за формулою (1.9).

4. Оновлення вагів прихованого шару за формулою (1.10).

5. Повторення циклу, поки не виконується умова збіжності.

Особливості використання схеми.

Представлена схема є ефективною лише для статичних об'єктів. Для динамічних систем необхідно розширювати вектор входів додатковими значеннями попередніх тактів, тобто формувати часові вибірки. У такому випадку в ролі нейрорегулятора також може виступати багатошаровий персептрон, а методика навчання залишається аналогічною.

Основна складність пов'язана з необхідністю визначення Якобіана об'єкта $f_0'(u(k))$, який зазвичай невідомий. Тому використовують його апроксимацію, наприклад:

$$f_0'(u(k)) \approx \frac{f_0(u(k))}{\Delta u(k)},$$

де

$$\Delta f_0(u(k)) = f_0(u(k-1)) - f_0(u(k-2)),$$

$$\Delta u(k) = u(k-1) - u(k-2).$$

1.2.2 Структура нейрокерування зі зворотним поширенням у часі

Альтернативний підхід до побудови систем нейроуправління передбачає створення нейромережевої моделі об'єкта (нейроемулятора), яка фактично виконує роль ідентифікатора динамічної системи. Функціонування такої системи відбувається у два етапи: спочатку нейрорегулятор навчається реалізації інверсної динаміки об'єкта, а на подальшому етапі ця модель використовується для організації процедури зворотного розповсюдження помилки під час реального керування. Використання нейроемулятора дає змогу визначати Якобіан систе-

ми, замінюючи безпосередній об’єкт його нейромережею-моделлю. У цьому разі маємо:

$$f_0'(u(k)) \approx \frac{\partial \hat{f}_{\text{c}}(u(k))}{\partial u(k)},$$

де $\hat{f}_{\text{c}}$ – відображення «вхід–вихід», що реалізується нейромережею-емулятором.

Підвищити швидкість навчання нейромережевої моделі об’єкта можна завдяки використанню комбінованого сигналу: на вихід об’єкта подається сума реакції деякої апріорної моделі $\hat{y}_1$ та коригувального сигналу нейромережі $\hat{y}_2$. У випадку невідповідності прогнозованому виходові виконується перенавчання, спрямоване на мінімізацію похибки ідентифікації. Таким чином, елемент нейроемулятора фактично є комбінацією «базової моделі» та коригувальної ШНМ (рис. 1.3), причому змінам підлягає лише нейромережева складова. Подібний підхід демонструє високу ефективність у задачах практичного керування. Найкращі результати забезпечують моделі типу AR, ARX, ARMAX, OE, BJ та інші структури, здатні виконувати довгострокове прогнозування, тоді як ШНМ компенсує нелінійні складові.

Узагальнена структура нейрокерування нелінійним динамічним об’єктом наведена на рис. 1.4. Тут ШНМ2 виконує роль ідентифікатора (рис. 1.5), за допомогою якого визначаються оцінки Якобіана, необхідні для навчання нейрорегулятора на основі ШНМ1, що реалізує інверсно-динамічний опис об’єкта.

Навчання нейрорегулятора здійснюється за принципом зворотного поширення помилки через нейромережеву модель об’єкта. Така конфігурація отримала назву «схема зворотного розповсюдження в часі» [48–70].

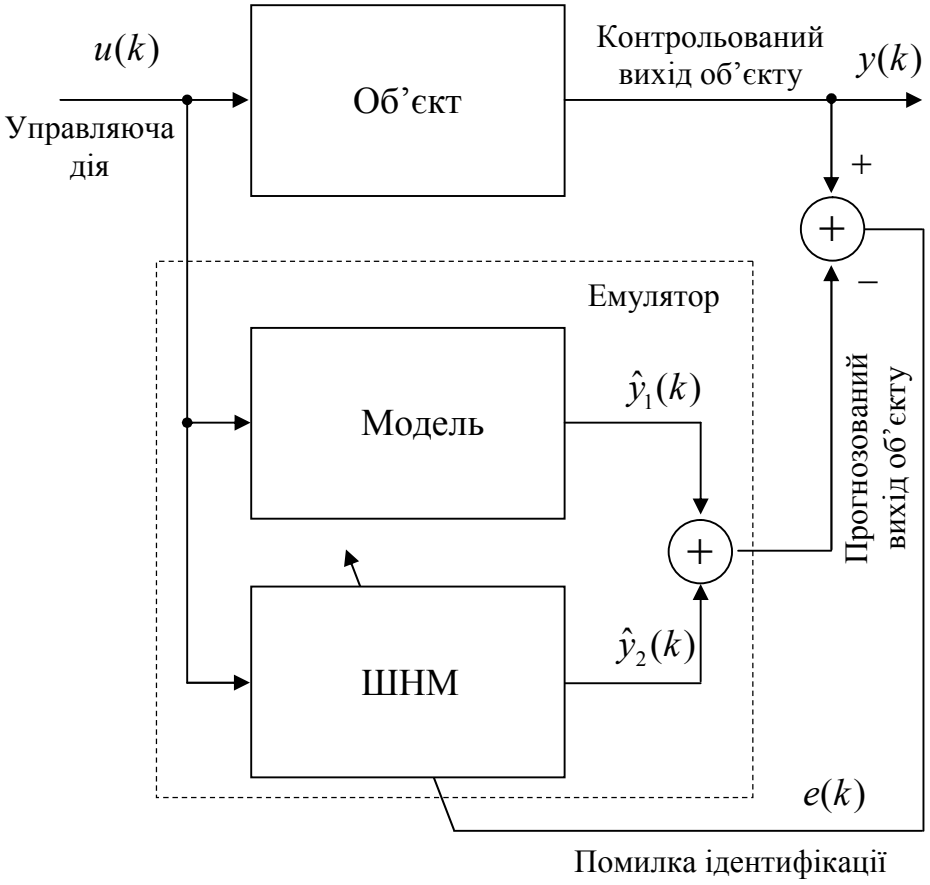


Рисунок 1.3 – Узагальнена структура нейромережевого емулятора із залученням допоміжної моделі

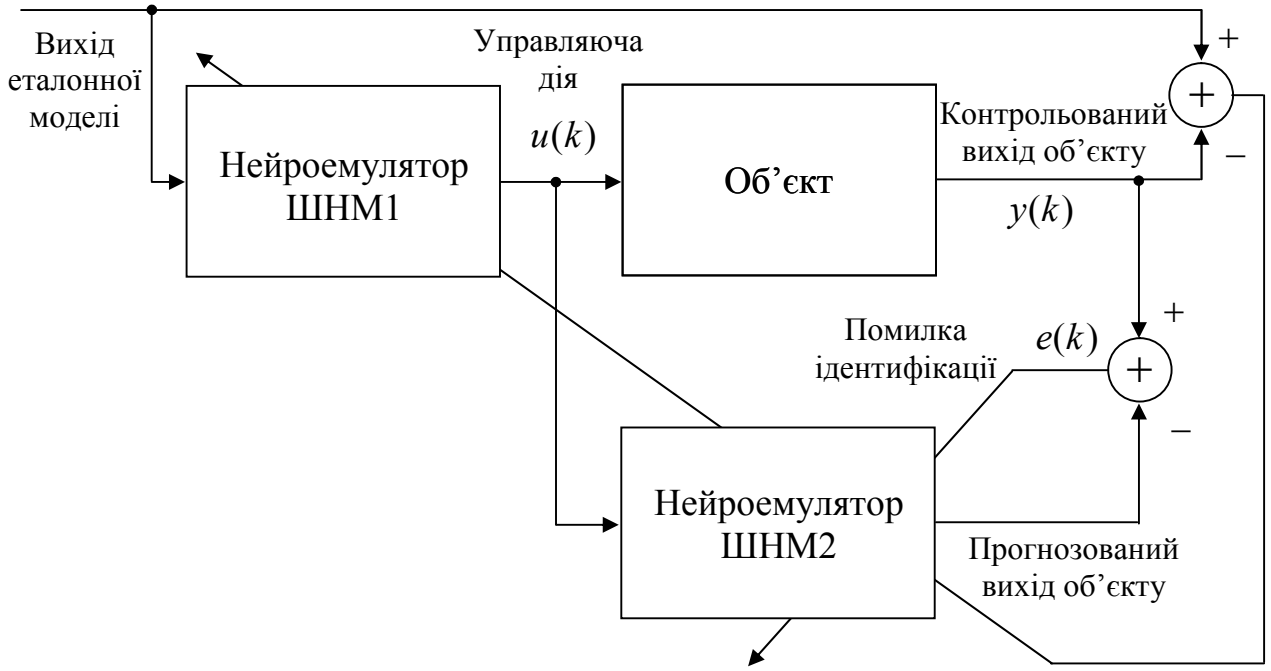


Рисунок 1.4 – Концептуальна схема інверсно-динамічного нейрокерування

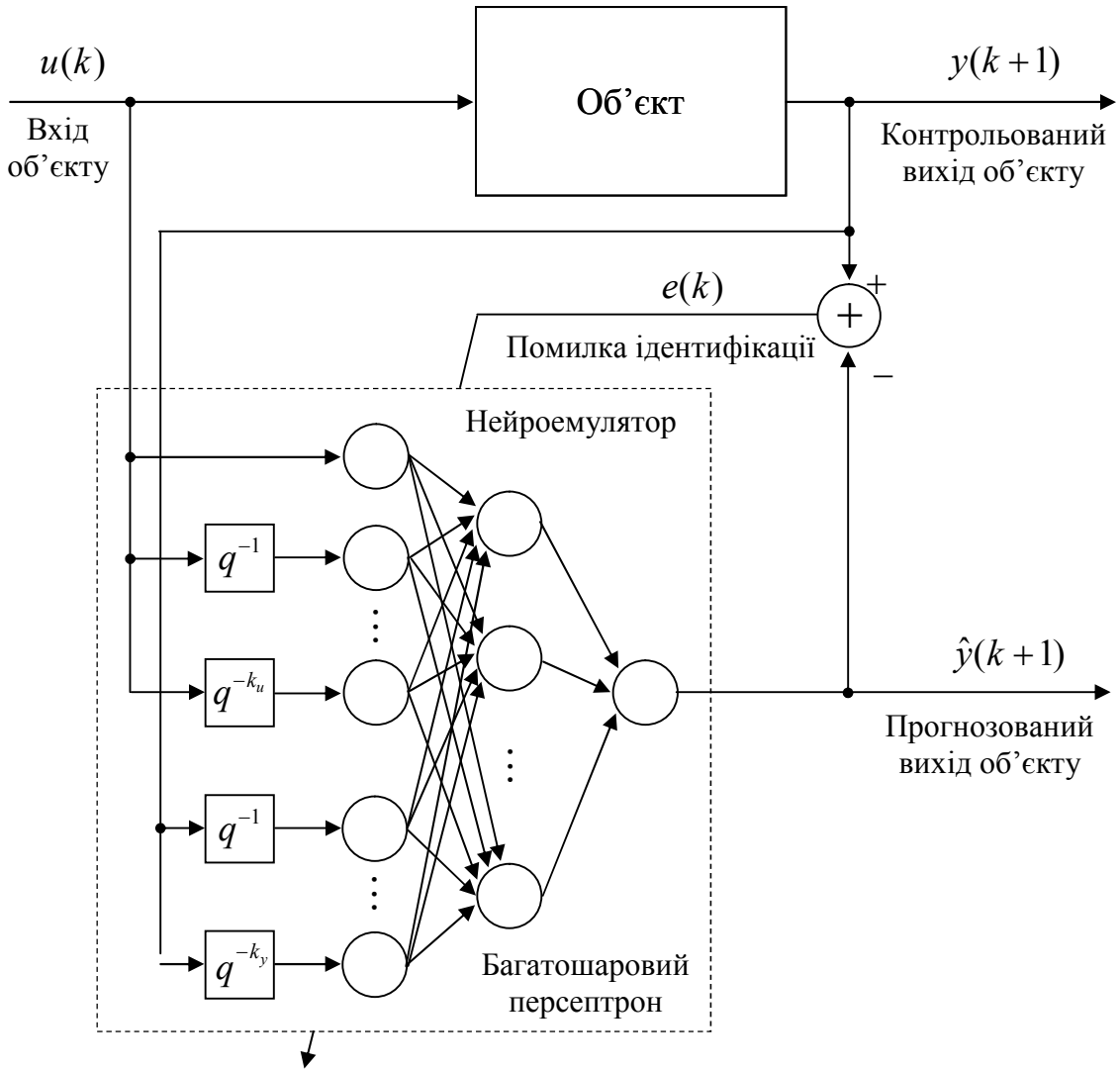


Рисунок 1.5 – Архітектура нейроемулятора для відтворення динаміки об’єкта

У роботі [71] запропоновано інший варіант побудови інверсної динаміки (рис. 1.6). У цій структурі нейроемулятор ШНМ2 спочатку навчається відтворенню інверсної динаміки об’єкта, після чого вага нейронів копіюється до нейрорегулятора ШНМ1. Процес копіювання моделі відбувається після завершення навчання ШНМ2. Само навчання може виконуватися як автономно, так і при подачі випадкових тестових сигналів на вхід реального об’єкта. У ролі структури нейроемулятора може бути використана схема, показана на рис. 1.7.

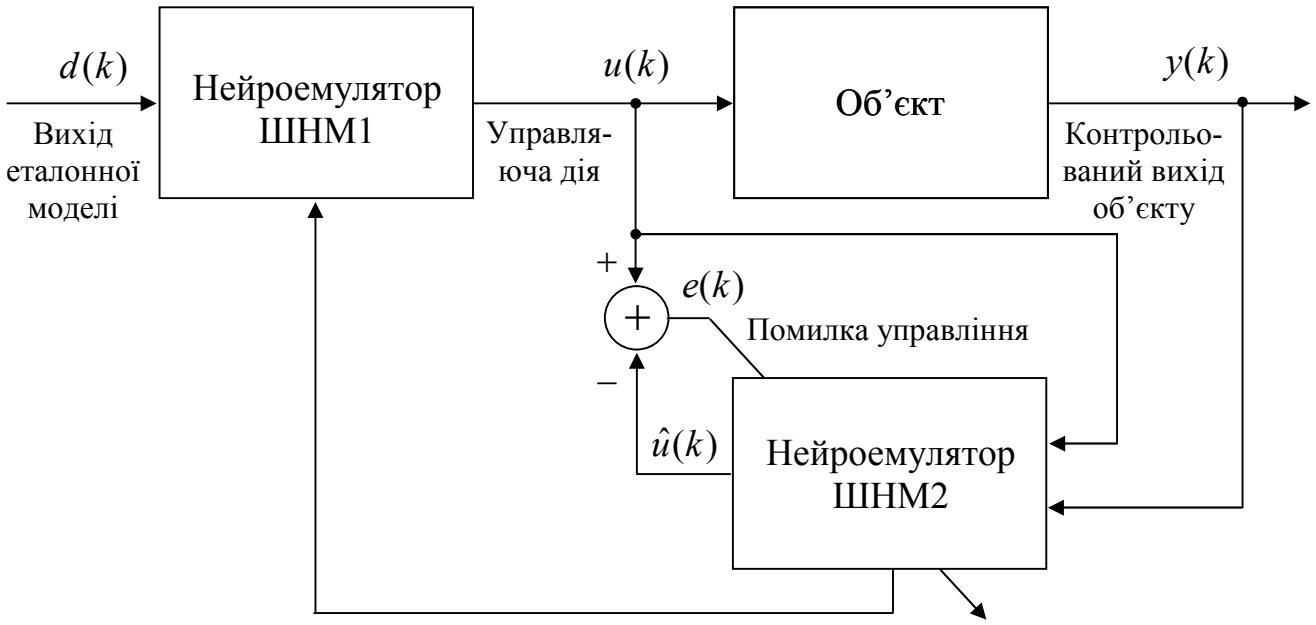


Рисунок 1.6 – Схема реалізації інверсного відображення на основі копіювання нейромоделі

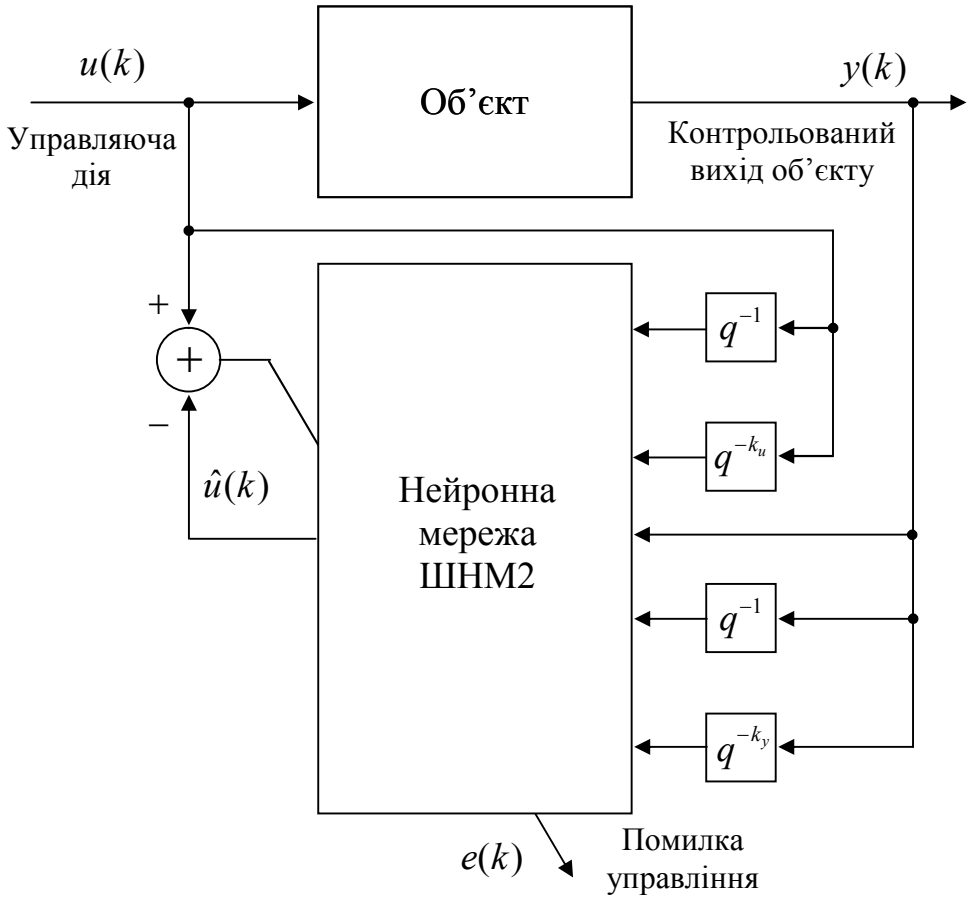


Рисунок 1.7 – Структурна модель нейронного емулятора, застосовуваного в схемі інверсного відображення

1.2.3 Паралельні структури нейронного керування

Одним із найпоширеніших підходів до реалізації інтелектуальних систем керування є паралельні схеми нейроуправління. У таких конфігураціях керувальний сигнал утворюється як суперпозиція виходів традиційного регулятора (наприклад, ПІД-регулятора) та нейромережевого регулятора (рис. 1.8). Роль нейрорегулятора полягає у корекції дії класичного регулятора, коли останній не здатен забезпечити необхідні показники точності або динамічної якості.

Одну з модифікацій паралельної схеми, що поєднує інверсну і пряму моделі об'єкта, було запропоновано в роботах [72, 73] (рис. 1.9).

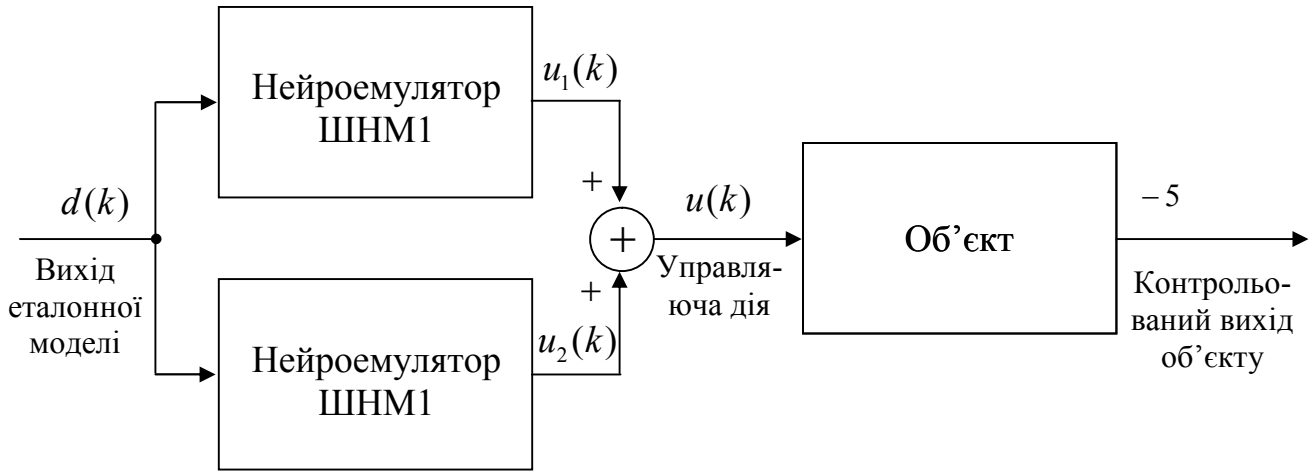


Рисунок 1.8 – Паралельна структура комбінованого класичного та нейромережевого керування

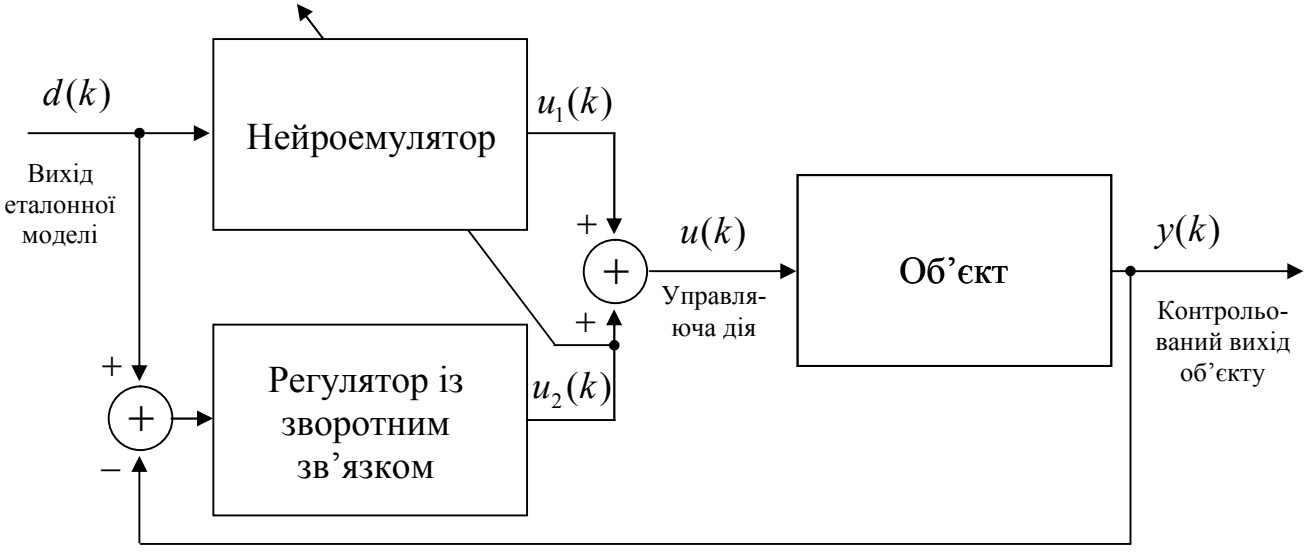


Рисунок 1.9 – Схема нейрокерування на основі помилки зворотного зв'язку

Налаштування параметрів нейромережевого регулятора виконується за методом навчання на основі помилки зворотного зв’язку. Для цього багаторазово повторюється цикл відтворення бажаної траєкторії до досягнення необхідної точності. Помилка регулювання при цьому надходить із виходу традиційного регулятора, підключеного паралельно нейромережі. Після завершення етапу навчання саме нейрорегулятор отримує функцію основного каналу керування, а класичний регулятор повністю вимикається з контуру.

Для узагальнення підходу розглянемо модифіковану структуру керування за помилкою зворотного зв’язку, наведеної на рис. 1.10 [74].

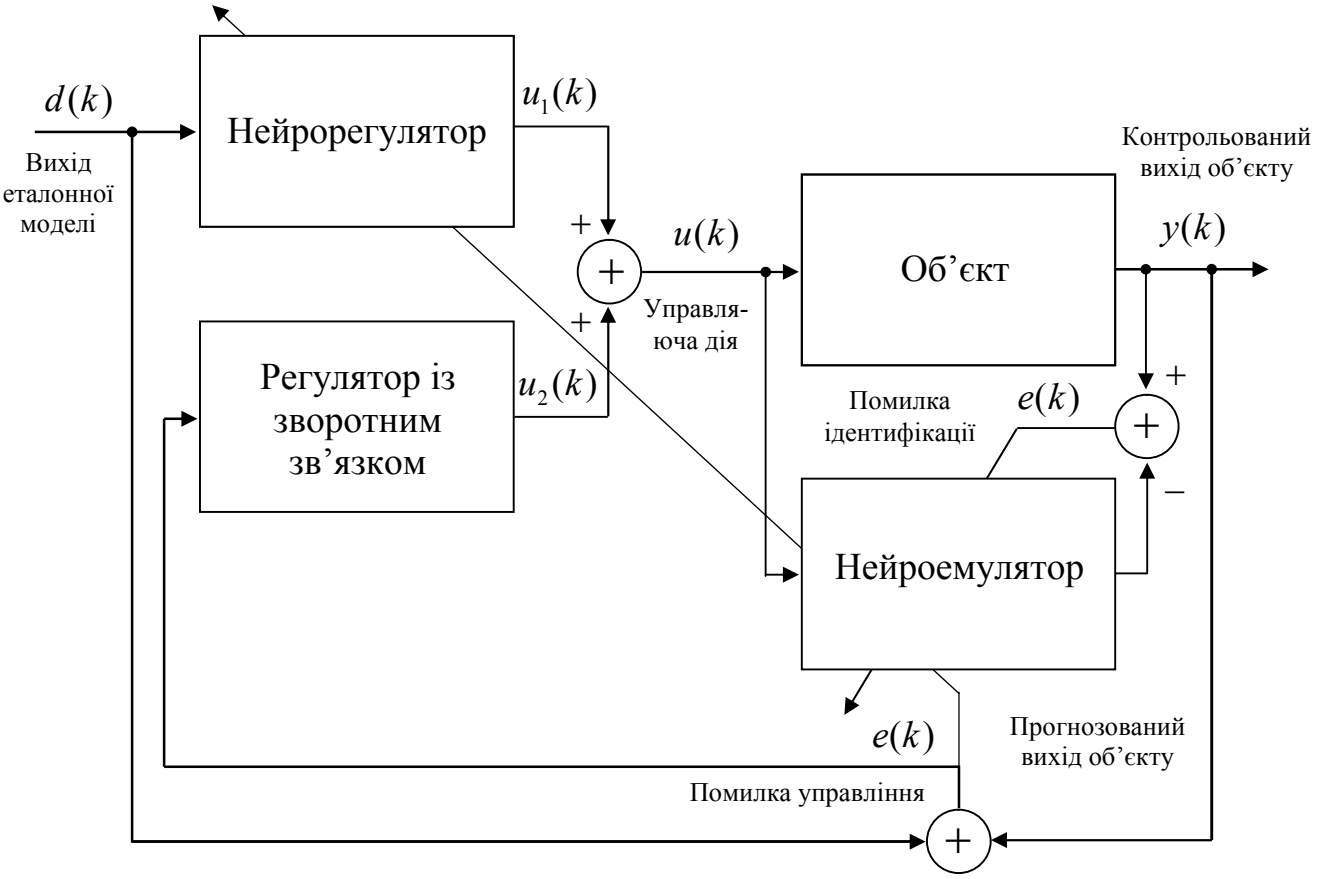


Рисунок 1.10 – Узагальнена схема нейронного керування, побудована на основі помилки зворотного зв’язку

Система керування складається з трьох основних компонентів.

1. Перший компонент (позначений жирними лініями) утворює традиційний контур зовнішнього зворотного зв’язку [75]. Він базується на помилці між

виходами еталонної моделі та реального об'єкта керування $e(k) = d(k) - y(k)$. У цій ланці зазвичай використовується ПІД-регулятор.

2. Другий компонент містить нейроемулятор, який моделює динамічну поведінку об'єкта. Його навчання проводиться на основі помилки ідентифікації $e(k) = y(k) - \hat{y}(k)$. Нейроемулятор формує внутрішній контур зворотного зв язку, що значно прискорює реакцію системи.

3. Третій компонент представлений нейрорегулятором, що наближає інверсну динаміку об'єкта. У процесі навчання задовольняють дві умови: нейроемулятор має відтворювати пряму модель, а нейрорегулятор — інверсну.

У період навчання визначальним є саме внутрішній контур, який поступово забезпечує переважний внесок у керування. Після завершення навчання традиційний регулятор повністю вимикається, а керування здійснюється нейромережею, яка забезпечує високу швидкодію та стійкість, навіть за умов наявності збурень.

Для виводу алгоритму навчання розглянемо об'єкт керування з часовою затримкою, описаний рівнянням:

$$\begin{aligned} y(k+1) &= f_0(u(k)) = f_0(u_1(k)) - u_2(k); \\ u_2(k) &= \Phi(k). \end{aligned} \tag{1.12}$$

Як і раніше, критерієм якості виступає квадратична функція помилки (1.3). Налаштування вагових коефіцієнтів виконується за методом найшвидшого градієнтного спуску. Архітектуру нейрорегулятора наведено на рис. 1.11.

Корекція ваг вихідного шару виконується за правилом:

$$\Delta w_{1j}^2(k+1) = -\eta \frac{\partial E}{\partial w_{1j}^2(k)},$$

а корекція ваг прихованого шару — за виразом

$$\Delta w_{j1}^1(k+1) = -\eta \frac{\partial E}{\partial w_{j1}^1(k)}.$$

Використовуючи локальну помилку вихідного шару $\delta^2 = \frac{\partial E}{\partial a_1^2}$ і рівняння (1.12), отримаємо:

$$\delta^2 = \frac{\partial E}{\partial e(k+1)} \frac{\partial e(k+1)}{\partial y(k+1)} \frac{\partial y(k+1)}{\partial u(k)} \frac{\partial u(k)}{\partial a_1^2} =$$

$$= e(k+1) \frac{\partial y(k+1)}{\partial u(k)} \left(\frac{\partial u_1(k)}{\partial a_1^2} + \frac{\partial u_2(k)}{\partial a_1^2} \right).$$

З урахуванням виду активаційної функції і значення $\frac{\partial u_2(k)}{\partial a_1^2} = 0$, маємо:

$$\delta^2 = (d(k+1) - y(k+1)) \frac{\partial y(k+1)}{\partial u(k)} u_2(k)(1 - u_2(k)). \qquad (1.13)$$

Тепер можемо записати $\Delta w_{1j}^2 = -\eta \delta^2 y_j^1$. Враховуючи, що робота *j*-го нейрона прихованого шару задається співвідношеннями (1.5) та (1.6), а також використовуючи визначення локальної помилки прихованого шару δ_j^1 з формула (1.13), правило корекції вагових коефіцієнтів прихованого шару набуває такого вигляду:

$$\Delta w_{j1}^1 = -\eta \delta_j^1 d,$$

де

$$\delta_j^1 = \delta^2 w_{1j}^2 y_j^1 (1 - y_j^1). \qquad (1.14)$$

Алгоритм навчання нейрорегулятора для паралельної схеми має такий вигляд:

1. Задати початкові значення вагових коефіцієнтів w та зміщень b, вибрати коефіцієнт навчання η.

2. На вхід мережі подати опорний сигнал $d(k)$ і обчислити локальні помилки δ^2 та δ_j^1 за формулами (1.13), (1.14).

3. Оновити ваги вихідного шару: $\Delta w_{1j}^2(k+1) = \Delta w_{1j}^2(k) - \eta \delta^2 y_j^1$.

4. Оновити ваги прихованого шару: $\Delta w_{j1}^1(k+1) = \Delta w_{j1}^1(k) - \eta^2 \delta_j^1 d$.

5. Повторювати кроки 2–4 до виконання критерія зупинки.

Як і у випадку послідовного нейрокерування, на кожному кроці навчання необхідно знати Якобіан об’єкта керування. Його можна апроксимувати або скористатися оцінками, отриманими нейроемулятором у процесі ідентифікації (рис. 1.11).

1.2.4 Пряме адаптивне нейроуправління

Розглянемо нелінійний об’єкт, динаміка якого описується NARX-моделлю

$$y(k+1) = f[y(k),\ y(k-1),...,y(k-k_y),u(k),\ u(k-1),...,u(k-k_u)] \qquad (1.15)$$

де $y(k)$ – вихід об'єкта;

$u(k)$ – керуючий сигнал;

$f(\cdot)$ – невідома нелінійна залежність;

k_u, k_y – – порядки запізнення за каналами управління та виходу .

Для побудови нейромережевої моделі об’єкта використовується двошаровий персептрон, вагові коефіцієнти якого коригуються за алгоритмом зворотного поширення помилки. Як активаційна функція нейронів прихованого шару застосовується гіперболічний тангенс:

$$f_{th}(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}},$$

а її похідна визначається виразом

$$f'_{th}(u) = 1 - f_{log}^2(u).$$

Для нейрона вихідного шару обирається лінійна функція активації. Введемо позначення для вхідного вектора нейроемулятора:

$$p = [y(k),\ y(k-1),...,y(k-k_y),u(k),\ u(k-1),...,u(k-k_u)].$$

Тоді нейромережева модель об'єкта (НММ) набуває вигляду

$$\hat{y}(k+1) = \hat{f}[y(k),\ y(k-1),...,y(k-k_y),u(k),\ u(k-1),...,u(k-k_u)], \qquad (1.16)$$

де $\hat{y}(k+1)$ – вихід нейромережі;

$\hat{f}[\cdot]$ – оцінка невідомої функції $f[\cdot]$.

Оскільки навчання нейромережі мінімізує квадрат помилки ідентифікації $e^2(k+1) = [y(k+1)) - \hat{y}(k+1)]^2$, то сигнал $\hat{y}(k+1)$ може використовуватись як прогнозоване значення виходу об'єкта (1.15). Це дозволяє сформулювати задачу управління як мінімізацію різниці між виходами еталонної моделі та НММ. Таким чином, критерій якості визначимо як

$$J = \frac{1}{2}e^2(k+1), \qquad (1.17)$$

де

$$e(k+1) = d(k+1) - \hat{y}(k+1). \qquad (1.18)$$

Керуючий сигнал $u(k)$ повинен забезпечувати мінімум функціоналу (1.17).

Враховуючи структуру мережі (1.16), її вихід можна записати у вигляді

$$\hat{y}(k+1) = w^{2^T}\ [\tanh(\mathbf{W^1}p + b^1)] + b^2, \qquad (1.19)$$

де $\mathbf{W^1}$ – матриця ваг прихованого шару;

w^2 – вектор ваг вихідного шару;

b^1 – вектор зміщень прихованого шару;

b^2 – зміщення вихідного нейрона.

Для мінімізації функціоналу (1.17) використаємо метод найшвидшого спуску:

$$u(k+1) = u(k) + \eta\frac{\partial J}{\partial u(k)}, \qquad (1.20)$$

де $\eta > 0$ – коефіцієнт навчання.

Очевидно, точність регулятора визначається якістю НММ, тому бажано, щоб $\hat{y}(k+1)$ асимптотично прагнув до $y(k+1)$. Це досягається шляхом on-line навчання моделі в процесі роботи.

Диференціюючи (1.17) за $u(k)$, маємо:

$$\frac{\partial J}{\partial u(k)} = -e(k+1)\frac{\partial \hat{y}(k+1)}{\partial u(k)}, \qquad (1.21)$$

де $\frac{\partial \hat{y}(k+1)}{\partial u(k)}$ – чутливість НММ до зміни керуючого сигналу $u(k)$.

Підставляючи (1.21) у (1.20), отримаємо

$$u(k+1) = u(k) + \eta e(k+1)\frac{\partial \hat{y}(k+1)}{\partial u(k)}.$$

Враховуючи структуру (1.19), градієнт обчислюється так:

$$\frac{\partial \hat{y}(k+1)}{\partial u(k)} = w^{2^T}\ [\mathrm{sech}(\mathbf{W^1}p + b^1)]^{-1}\mathbf{W^1}\frac{\partial p}{\partial u(k)}$$

де $\frac{\partial p}{\partial u(k)} = [0,0,...,0,1,0,...,0]^T$.

Тоді остаточний вираз для керуючого сигналу набуває вигляду

$$u(k+1) = u(k) + \eta e(k+1)w^{2T}[\mathrm{sec\,h}^2(\mathbf{W^1}p + b^1)]^{-1}\mathbf{W^1}\frac{\partial p}{\partial u(k)}. \qquad (1.22)$$

Алгоритм прямого адаптивного нейроуправління.

1. Обчислення виходу нейромережевої моделі $\hat{y}(k+1)$ за (1.19).
2. Обчислення помилки управління $e(k+1)$ за (1.18).
3. Корекція ваг нейромережі за алгоритмом зворотного поширення помилки.
4. Обчислення управляючого сигналу $u(k+1)$ відповідно до (1.22).
5. Подача $u(k+1)$ на вхід об’єкта.
6. Повторення кроків 1–5.

Якість роботи алгоритму можна підвищити, застосовуючи методи прогнозного управління, де враховуються не лише поточні, а й передбачені значення вхідних і вихідних сигналів об'єкта. Оскільки нейромережеву модель (1.19) можна вважати асимптотично еквівалентною реальному об'єкту, вона здатна виконувати функцію прогнозування майбутніх значень вихідного сигналу.

1.2.5 Управління з самонастройкою

Одним із напрямків застосування штучних нейронних мереж у задачах управління є використання схем нейроуправління з **самонастройкою**. У такому випадку ШНМ виконує функції оператора системи, тобто автоматично підбирає і налаштовує параметри класичного регулятора. Багато методів адаптивного управління передбачають наявність параметрів, які зазвичай вибираються вручну або підбираються методом проб і помилок. Використання ШНМ дозволяє автоматизувати процес підбору та налаштування параметрів традиційних регуляторів, що робить нейроуправління з самонастройкою перспективним для багатьох технологічних процесів.

На рис. 1.11 показано приклад схеми нейроуправління з самонастройкою для ПІД-регулятора.

ШНМ використовується для автоматичної настройки коефіцієнтів ПІД-регулятора з метою мінімізації помилки управління. Закон функціонування ПІД-регулятора в дискретному часі описується різницевим рівнянням:

$$u(k) = u(k-1) + K_P(e(k) - e(k-1)) + K_I e(k-1) + \\ + K_D(e(k) - 2\,e(k-1) + eu(k-2)), \tag{1.23}$$

де K_P, K_I, K_D – пропорційний, інтегральний і диференціальний коефіцієнти відповідно;

$e(k) = d(k) - y(k)$ – помилка управління;

$d(k)$ – задане значення (уставка).

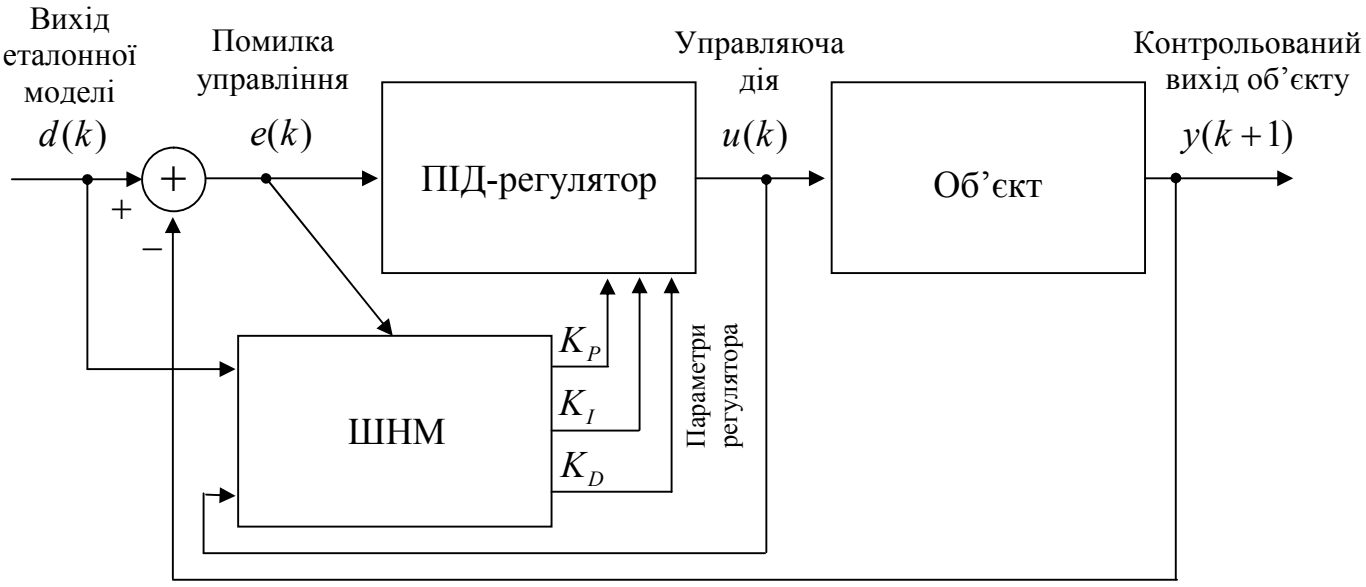


Рисунок 1.11 – Структура нейроуправління
з самонастройкою ПІД-регулятора

Розглянемо об’єкт управління, динаміка якого описується відповідним $y(k+1) = f_0(u(k))$. Для оцінки ефективності управління в якості критерію якості застосовується квадратична функція помилки

$$E = \frac{1}{2} e^2(k+1). \tag{1.24}$$

Для автоматичної настройки ПІД-коефіцієнтів побудуємо двошаровий персептрон з трьома нейронами у вихідному шарі. При цьому кожний вихід ШНМ відповідає певному коефіцієнту ПІД-регулятора:

$$K_P = y_1^2,\ K_I = y_2^2,\ K_D = y_3^2. \tag{1.25}$$

Для реалізації нелінійних відображень нейронів прихованого шару застосовуються логістичні функції активації, а для нейронів вихідного шару – лінійні функції, щоб забезпечити необхідний діапазон значень коефіцієнтів. Вагові коефіцієнти ШНМ налаштовуються за алгоритмом найшвидшого спуску.

Для нейронів вихідного шару градієнт квадратичної функції помилки визначається як:

$$\frac{\partial E}{\partial w_{ij}^2} = \frac{\partial E}{\partial e(k+1)} \frac{\partial e(k+1)}{\partial y(k+1)} \frac{\partial y(k+1)}{\partial u(k)} \frac{\partial u(k)}{\partial y_i^2(k)} \frac{\partial y_i^2(k)}{\partial a_i^2(k)} \frac{\partial a_i^2(k)}{\partial w_{ij}^2(k)}.$$

Оскільки $e(k+1) = d(k+1) - y(k+1)$ з урахуванням (1.24) можна записати

$$\frac{\partial E}{\partial e(k+1)} \frac{\partial e(k+1)}{\partial y(k+1)} = -(d(k+1) - y(k+1) = -e(k+1).$$

Правило корекції ваг вихідного шару записується наступним чином:

$$y_i^2 = a_i^2 = \sum_{l=1}^{N^{L-1}} w_{il}^2 y_l^1 + b_{i0}^2, \quad i = 1,\ 2,\ 3;$$

$$\frac{\partial y_i^2(k)}{\partial a_i^2(k)} = 1, \quad \frac{\partial a_i^2(k)}{\partial w_{ij}^2(k)} = y_j^1, \quad i = 1,\ 2,\ 3; \quad j = 1,\ 2,..., N^1. \qquad)$$

Спираючись на закон функціонування ПІД-регулятора (1.23) та враховуючи визначення виходів нейромережі (1.25), складаємо рівняння для обчислення часткових похідних $\frac{\partial u(k)}{\partial y_i^2(k)}$:

$$\frac{\partial u(k)}{\partial y_1^2(k)} = e(k) - e(k-1),$$

$$\frac{\partial u(k)}{\partial y_2^2(k)} = e(k-1),$$

$$\frac{\partial u(k)}{\partial y_3^2(k)} = e(k) - 2e(k-1) + e(k-2).$$

Для нейронів прихованого шару локальна помилка обчислюється як:

$$\delta_i^2 = e(k+1)\frac{\partial y(k+1)}{\partial u(k)}\frac{\partial u(k)}{\partial y_i^2(k)}. \qquad (1.26)$$

Правило оновлення ваг вихідного шару:

$$w_{ij}^2(k+1) = w_{ij}^2(k) - \eta\delta_i^2 y_j^1, \quad i = 1,\ 2,\ 3;\ \ j = 1,\ 2,..., N^1. \qquad (1.27)$$

Правило оновлення ваг прихованого шару визначається формулою:

$$\frac{\partial E}{\partial w_{ij}^1} = \sum_{n-1}^{3}\frac{\partial E}{\partial e(k+1)}\frac{\partial e(k+1)}{\partial y(k+1)}\frac{\partial y(k+1)}{\partial u(k)}\frac{\partial u(k)}{\partial y_n^2(k)}\frac{\partial y_n^2(k)}{\partial a_n^2(k)}\frac{\partial a_n^2(k)}{\partial y_j^1(k)}\frac{\partial y_j^1(k)}{\partial a_j^1(k)}\frac{\partial a_j^1(k)}{\partial w_{ij}^1(k)} =$$

$$= \sum_{n-1}^{3} e(k+1)\frac{\partial y(k+1)}{\partial u(k)}\frac{\partial u(k)}{\partial y_n^2(k)} w_{jn}^2 y_j^1(k)(1-y_j^1(k))d_j(k) = -\delta_i^1 d_j(k),$$

де δ_i^1 – локальна помилка прихованого шару

$$\delta_i^1 = \sum_{n-1}^{3} e(k+1)\frac{\partial y(k+1)}{\partial u(k)}\frac{\partial u(k)}{\partial y_n^2(k)} w_{jn}^2 y_j^1(k)(1-y_j^1(k)) =$$

$$= y_j^1(k)(1-y_j^1(k))\sum_{n-1}^{3}\delta_i^1 w_{jn}^2. \qquad (1.28)$$

В результаті отримуємо правило корекції вагових коефіцієнтів нейронів прихованого шару у такій формі:

$$w_{ij}^1(k+1) = w_{ij}^1(k) - \eta\delta_i^1 d_j, \quad i = 1,\ 2,..., N^1;\ \ j = 1,\ 2,..., N^0 \qquad (1.30)$$

Для роботи алгоритму на кожному такті необхідно обчислювати Якобіан об'єкта управління $\frac{\partial y(k+1)}{\partial u(k)}$, що може бути реалізовано за допомогою нейроемулятора.

Алгоритм настройки вагів ПІД-нейрорегулятора з самонастройкою

1. Ініціалізуються ваги w та зсуви b, задається швидкість навчання η, встановлюється $k = 0$.

2. На вхід ШНМ подається опорний сигнал $d(k)$; обчислюються помилка управління $e(k+1)$ та локальні помилки вихідного δ_i^2 і прихованого δ_i^2 шарів за формулами (1.26) та (1.28).

3. Оновлюються ваги вихідного шару за формулою (1.27).

4. Оновлюються ваги прихованого шару за формулою (1.30).

5. Повторюються кроки 2–4 до досягнення заданої точності управління.

1.2.6 Нелінійне предиктивное управління

Суть предиктивного методу регулювання на основі моделі (Model Predictive Control, MPC) полягає у формуванні послідовності управляючих сигналів таким чином, щоб мінімізувати розбіжність між бажаним значенням вихідного сигналу об’єкта та прогнозованим виходом його моделі на майбутньому горизонті часу (позначеному пунктирними лініями на рис. 1.12) [76–80].

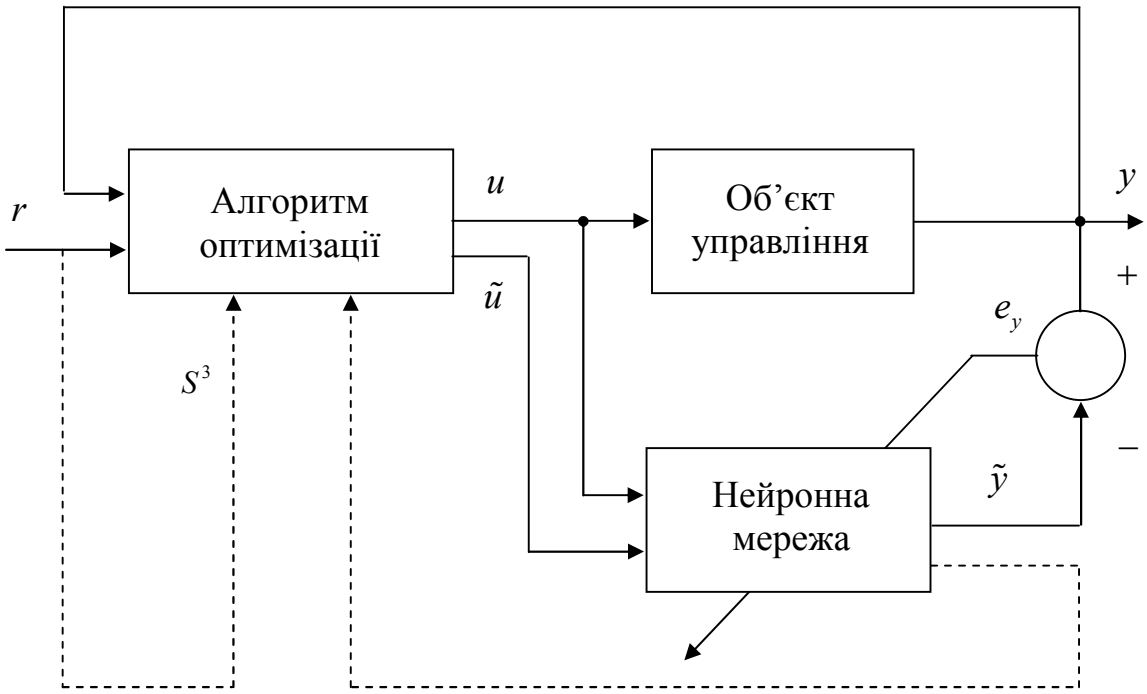


Рисунок 1.12 – Схема предиктивного регулювання (MPC), що показує прогнозовані вихідні сигнали на горизонті прогнозу

У цьому підході регулятор реалізується як ітераційний алгоритм оптимізації, який включає такі ключові кроки:

- мінімізація обраного функціоналу якості здійснюється чисельними методами оптимізації;
- оптимізується функціонал, що відображає бажані критерії якості регулювання;
- для фактичного впливу на об’єкт використовується лише перший елемент оптимізованої послідовності сигналів;
- процедура повторюється на кожному кроці дискретності, що дозволяє враховувати змінні умови процесу.

Головна перевага методу MPC полягає у забезпеченні високої якості регулювання навіть для складних нелінійних об’єктів. Водночас, недоліком цього підходу є значна обчислювальна складність і підвищені вимоги до точності моделі об’єкта.

У рамках даної магістерської роботи застосовується саме нелінійне предиктивне регулювання, що дозволяє ефективно керувати об’єктами з вираженими нелінійними характеристиками.

1.3 Вимоги до електроустаткування мостових кранів

Сучасні промислові підприємства все більше впроваджують високий рівень механізації та автоматизації виробничих процесів. Це стосується не лише основних технологічних операцій, але й численних допоміжних завдань, таких як транспортування сировини, готової продукції, палива та матеріалів. У багатьох випадках ці завдання ефективно виконуються за допомогою електричних кранів, що дозволяє зменшити ручну працю, підвищити продуктивність та безпеку робіт.

Електричні крани мають широке поширення у різних галузях промисловості. Наприклад:

- Мостові крани застосовуються у цехах металургійних та машинобудівних заводів;

- Портальні та козлові крани використовуються на рудних дворах, складах вугілля електростанцій та великих перевантажувальних майданчиках;
- Баштові, кабельні та інші крани – у будівельній сфері та для робіт на висоті.

Окрім стаціонарних кранів, велике значення мають пересувні крани, зокрема автомобільні, гусеничні, залізничні та плавучі. Кожна з цих категорій включає численні конструктивні модифікації, адаптовані під специфіку об'єкта або технологічного процесу. Наприклад, мостові крани можуть бути оснащені різними вантажозахоплювальними пристроями: крюками, грейферами, кліщами, посадочними механізмами тощо.

Серед промислових кранів особливо широке застосування має таунтовий (мостовий) кран, який є універсальним підйомно-транспортним пристроєм. Він складається зі сталевої мостової конструкції, що спирається на ходові візки і переміщується по підкранових рейках, закріплених на стаціонарних опорах над обслуговуваною площею. Вздовж моста рухається візок з підйомною лебідкою, яка забезпечує підйом і опускання вантажів. Основними механізмами мостового крана є:

- механізм переміщення моста;
- механізм пересування візка;
- підйомна лебідка.

Всі ці механізми оснащені власними електроприводами, які повинні забезпечувати стабільну та надійну роботу навіть у повторно-короткочасному режимі з великою частотою включень. Електроустаткування мостових кранів повинно функціонувати в умовах підвищеної запиленості, високої вологості повітря, значних температурних коливань та інших несприятливих факторів виробничого середовища.

Крім того, до електричного оснащення кранів висуваються суворі вимоги щодо:

- безперебійності роботи;
- високої продуктивності;

- безпеки обслуговування;
- зручності експлуатації та обслуговування;
- енергозбереження та ефективного використання електроприводів.

Сучасні тенденції передбачають також впровадження систем автоматизації та цифрового контролю, інтеграцію з SCADA-системами, дистанційне керування та моніторинг стану механізмів у реальному часі. Це дозволяє підвищити надійність роботи мостових кранів, оптимізувати витрати електроенергії та мінімізувати ризики аварійних ситуацій.

1.4 Обґрунтування вибору системи електроприводу механізму переміщення моста мостового крана

При виборі електроприводу для механізму переміщення моста мостового крана враховуються такі ключові параметри:

- вантажопідйомність і швидкість руху крана;
- діапазон регулювання швидкості та жорсткість характеристики приводу;
- конструктивні параметри та маса механізмів;
- інтенсивність експлуатації, тобто кількість включень на годину та тривалість роботи;
- умови навколишнього середовища, включаючи температуру, вологість та запиленість;
- експлуатаційні витрати, у тому числі витрати на електроенергію та технічне обслуговування.

Економічна ефективність системи оцінюється за принципом мінімізації загальних витрат, що включають:

- первісні інвестиції;
- витрати на експлуатаційне обслуговування та ремонт;
- витрати на електроенергію під час пуску, гальмування та тривалої роботи крана.

При цьому вибір проводиться з урахуванням повного циклу експлуатації крана (до капітального ремонту, приблизно 10 років). Якщо економічні показники порівнюваних систем близькі, додатково враховуються габаритні та масові характеристики приводу, а також умови розміщення електроустаткування на крані. В окремих випадках доцільно віддати перевагу системі з трохи меншими економічними показниками, але зручнішою у монтажі та експлуатації.

Тиристорні електроприводи постійного струму (ТП-Д) використовуються на кранах, коли необхідне точне регулювання швидкості при потужностях понад 60 кВт. За характеристиками регулювання вони подібні до систем із генератором-послідовником (Г-Д), проте мають суттєві переваги, що дозволили їм витіснити старі системи Г-Д. Сьогодні тиристорні приводи застосовуються на потужних мостових кранах ливарних цехів, перегружателях, бетоноукладачах, високопродуктивних баштових кранах для висотного будівництва та інших складних кранівських комплексах. Потужність таких приводів може досягати 400–600 кВт.

ТП-приводи розробляються на основі стандартних схем, які залежать від типу використаного тиристорного перетворювача (ТП):

- реверсивні ТП серій АТРК і ТПЕ;
- нереверсивні ТП серії АТК та з контакторним реверсом.

Нереверсивні ТП доцільні для приводів потужністю до 100 кВт і з невеликою кількістю включень при режимі роботи до 4М. Для більш інтенсивних режимів і потужностей понад 100 кВт використовуються реверсивні схеми.

Залежно від призначення механізму, тиристорні електроприводи можуть бути одно- або багатодвигунними. Багатодвигунні системи зазвичай застосовуються для механізмів пересування крана, а для механізму підйому – при високій потужності системи, найчастіше з двома двигунами. При цьому електричні схеми одно- та багатодвигунних систем істотних відмінностей не мають.

Після проведення техніко-економічного аналізу для механізму переміщення моста мостового крана обирається електропривід за схемою «тиристор-

ний перетворювач – двигун», який забезпечує оптимальне співвідношення точності управління, надійності та економічної ефективності.

1.5 Постановка задачі дослідження

Синтез та дослідження динамічних характеристик електроприводу механізму переміщення моста мостового крана зазвичай проводяться з припущенням про нескінченну жорсткість всіх з'єднань механічної частини системи. Однак таке спрощення відображає середню поведінку елементів і не враховує реальні пружні властивості зв'язків. Через кінцеву жорсткість елементів механічна частина електроприводу є пружною багатомасовою системою: прикладання управляючого моменту двигуна або зовнішнього навантаження викликає коливання мас, що підвищує максимальні навантаження на з'єднання та знижує точність переміщень. Традиційні регулятори не завжди забезпечують необхідні показники якості управління в таких умовах.

Аналіз сучасної літератури показує, що існує багато підходів до синтезу систем управління, але універсального рішення поки не розроблено. Наприклад, при керуванні механізмом повороту стріли роторного екскаватора також необхідно враховувати багатомасову пружність опорно-поворотного механізму та коливання, що виникають під дією як управляючих, так і зовнішніх збурюючих сил.

Використання нейромережевих технологій управління суттєво спрощує математичний аналіз та синтез систем, оскільки характеристики керування визначаються фундаментальними властивостями багатошарових нелінійних нейронних мереж, а не аналітично виведеними законами управління. Налаштовувані багатошарові нейронні мережі дозволяють ефективно вирішувати складні нелінійні задачі управління динамічними об'єктами.

Апроксимаційні властивості нейромереж грають ключову роль у формуванні нелінійних алгоритмів керування. Згідно з теоремою Стоуна–Вейерштрасса, багатошарові нейронні мережі здатні з довільною точністю ап-

роксимувати будь-яку безперервну функцію багатьох змінних на обмеженій множині, що теоретично обґрунтовує можливість використання нейромереж для формування оптимальних функцій управління в динамічних системах.

Завдяки своїй структурі та алгоритмам навчання, багатошарові мережі здатні формувати управляючі сигнали будь-якої форми, що робить їх ефективним інструментом для управління складними нелінійними системами. Адаптивність нейромереж дозволяє коригувати закони управління в реальному часі при неконтрольованих змінах динамічних та статичних характеристик об’єкта, використовуючи поточні вимірювані сигнали.

Паралельна обробка сигналів в нейронних мережах робить їх природним вибором для управління багатовимірними та багатоканальними системами, що робить багатошарові нейронні мережі універсальним обчислювальним середовищем для побудови високоефективних регуляторів нелінійних динамічних об’єктів.

Аналіз сучасних досліджень свідчить, що напрямок застосування нейромереж для керування механізмами з пружними кінематичними зв’язками є перспективним та актуальним.

Мета роботи – розробка нейромережевої системи управління механізмом переміщення моста мостового крана з урахуванням пружності механічних зв’язків.

Для досягнення мети необхідно вирішити наступні завдання:

1. Обрати систему електроприводу механізму переміщення моста, включаючи електродвигун та основне силове електроустаткування, провести синтез системи управління і моделювання на ЕОМ.

2. Розробити математичну модель системи з урахуванням кінцевої жорсткості з’єднань між двигуном та мостом, провести моделювання багатомасової системи і проаналізувати результати.

3. Розробити структурну схему нейромережевої системи управління, що забезпечує високоякісне регулювання з урахуванням пружності механічних зв’язків.

4. Синтезувати модель багатомасової системи за допомогою багатошарової прямонаправленої мережі з високою точністю ідентифікації.

5. Використовуючи принцип предиктивного управління на основі нейромережевої моделі об’єкта, розробити нейромережевий регулятор з високою швидкодією та якістю управління.

6. Провести моделювання нейромережевої системи та виконати аналіз отриманих результатів.

ВИСНОВКИ ДО РОЗДІЛУ 1

1. Аналіз літературних джерел підтвердив доцільність застосування нейронних мереж у задачах управління складними динамічними об’єктами. Сучасні системи управління повинні ефективно адаптуватися до змінних умов роботи за рахунок оперативного коригування параметрів та структури законів управління. Одним із ефективних підходів до реалізації таких вимог є використання методів нейромережевого моделювання та управління, що дозволяє виконувати як попередню ідентифікацію та налаштування системи, так і її адаптивне регулювання під час експлуатації.

2. Для задач управління доцільно застосовувати багатошарові нейронні мережі персептронного типу, включно з варіантами з зворотними зв’язками та вхідними лініями затримки по сигналам об’єкта. Використання динамічних алгоритмів навчання таких мереж дозволяє створювати адаптивні системи управління, які забезпечують стабільну та ефективну роботу складних об’єктів навіть за умов невизначеності та коливань навантажень.

3. Для оптимальної роботи об’єкта необхідне коригування параметрів регуляторів у відповідності до змін робочих умов. Це робить актуальним поєднання адаптивного підходу з методами штучних нейронних мереж, що дозволяє будувати прості моделі об’єкта та забезпечувати їх динамічне переналаштування разом із параметрами регулятора.

4. На основі аналізу сучасних структур нейрорегулювання обрано принцип нелінійного предиктивного регулювання для побудови нейромережевого регулятора системи, що дозволяє прогнозувати майбутні дії об’єкта та підвищувати точність і швидкодію управління.

5. Проведено огляд методів опису нелінійних динамічних об’єктів та принципів побудови нейромережевих моделей NARMAX та NARX. Моделі типу NARX параметризуються лінійно, що дає змогу застосовувати стандартні методи тренування нейромереж і забезпечує їх ефективність при ідентифікації систем.

6. Аналіз характеристик мостових кранів із урахуванням пружності механічних зв’язків підтвердив доцільність використання нейромережевих технологій для розробки систем управління механізмом переміщення моста.

7. Сформульована задача дослідження полягає у створенні нейромережевої системи управління механізмом переміщення моста мостового крана, яка враховує податливість механічних зв’язків та забезпечує високу точність, швидкодію та адаптивність управління.

РОЗДІЛ 2
РОЗРАХУНОК СИСТЕМИ АВТОМАТИЧНОГО УПРАВЛІННЯ МЕХАНІЗМОМ ПЕРЕМІЩЕННЯ МОСТА МОСТОВОГО КРАНА

2.1 Технічні характеристики мостового крана. Вимоги до системи управління

Вихідні дані для розрахунку електроприводу механізму переміщення мостамостового крана і синтезу системи управління наведені в табл. 2.1.

Таблиця 2.1 – Технічні дані мостового крана

Параметри	Величина
Вантажопідйомність $G_{вп}$, *кг*	37000
Продуктивність крана Q , *кг/с*	55
Маса моста(без візка) $m_м$, *кг*	28000
Маса візка $m_в$, *кг*	8000
Діаметр коліс моста $D_м$, *м*	0,71
Діаметр коліс візка $D_в$, *м*	0,4
Диаметр цапф коліс моста $d_м$, *м*	0,145
Диаметр цапф коліс візка $d_в$, *м*	0,095
Швидкість пересування моста $V_м$, *м/с*	1,25
Швидкість пересування візка $V_в$, *м/с*	0,65
Швидкість підйому $V_п$, *м/с*	0,1
Маса вантажозахватного пристрою m_0, *кг*	1000
Середня висота підйому і спуску $L_п$, *м*	6
Середній шлях моста в одну сторону $L_м$, *м*	45
Середній шлях візка в одну сторону $L_в$, *м*	24
Діаметр барабана механізму підйому D, *м*	0,65

Вимоги до показників якості функціонування системи управління.

Система управління повинна бути астатичною. Перерегулювання швидкості при ступінчастій вхідній дії не більше 10%, час регулювання не більше 10 с.

2.2 Оцінювання необхідної потужності електроприводу та добір електродвигунів

Статична потужність, яку повинен розвивати електродвигун механізму пересування моста, визначається за наступним співвідношенням:

$$P_{\text{ст м}} = \frac{k_p g (G_{\text{вп}} + m_{\text{мех}}) V_{\text{м}}}{\eta_{\text{н м}} R_{\text{м}}} \left(\frac{\mu d_{\text{м}}}{2} + f \right),$$

де $k_{\text{р}} = 1,3$ – коефіцієнт тертя реборд коліс об поверхню рейки;

$m_{\text{мех}}$ – загальна маса моста разом із візком та вантажозахватним обладнанням;

$$m_{\text{м}\Sigma} = m_{\text{м}} + m_{\text{в}} + m_0,$$

$$m_{\text{м}\Sigma} = 28000 + 8000 + 1000 = 37000 \ кг,$$

$\mu = 0,015$ – коефіцієнт втрат у підшипниках;

$f = 0,5 \cdot 10^{-3} \ м$ – коефіцієнт кочення ходових коліс;

$R_{\text{м}} = D_{\text{м}} / 2 = 0,355 \ м$ – радіус ходового колеса;

$\eta_{\text{н м}} = 0,82$ – ККД механізму при номінальному навантаженні.

Підставивши вихідні числові параметри у формулу, отримуємо:

$$P_{\text{ст м}} = \frac{1,3 \cdot 9,81 \cdot (37000 + 37000) \cdot 1,25}{0,82 \cdot 0,355} \cdot \left(\frac{0,015 \cdot 0,145}{2} + 0,5 \cdot 10^{-3} \right) \cdot 10^{-3} = 6 \ кВт.$$

Середня тривалість одного переміщення моста визначається як:

$$t_{\text{м}} = \frac{2 L_{\text{м}}}{V_{\text{м}}},$$

$$t_{\text{м}} = \frac{2 \cdot 45}{1,25} = 72 \ с.$$

Відносна тривалість роботи механізму пересування моста в межах робочого циклу обчислюється за виразом:

$$ПВ_{м} = \frac{t_{м}}{T} \cdot 100\%,$$

T – загальна тривалість робочого циклу

$$T = \frac{G_{вп}}{Q},$$

$$T = \frac{37000}{55} = 672\ c,$$

$$ПВ_{м} = \frac{72}{672} \cdot 100 = 10,7\%.$$

Розрахункова потужність електроприводу моста мостового крана визначається як:

$$P_{р\,м} = \frac{0,66(G_{вп} + m_{мех})a_{мех}V_{м}}{\eta_{н\,м}} + \frac{P_{ст\,м}}{1,75},$$

де $a_{мех} = 0,6 м/с^2$ – допустиме прискорення механізму

Після числової підстановки отримуємо:

$$P_{р\,м} = \frac{0,66 \cdot (37000 + 37000) \cdot 0,6 \cdot 1,25 \cdot 10^{-3}}{0,82} + \frac{6}{1,75} = 46,2\ кВт.$$

Щоб привести отриману потужність до стандартної (нормованої) тривалості включення, використовують наступну формулу:

$$P_{м} = P_{р\,м}\sqrt{\frac{ПВ_{м}}{ПВ_{ст}}},$$

$$P_{м} = 46,1 \cdot \sqrt{\frac{11,7}{40}} = 24,9\ кВт$$

На основі виконаних розрахунків та приймаючи до уваги вимоги до надійності та режиму роботи привода, для механізму переміщення моста доцільно застосувати два двигуни постійного струму типу Д-806. Технічні характеристики двигуна: $P_{\text{н}} = 16\ кВт$, $n_{н} = 710 об/хв$, $I_{\text{н}} = 84 A$, $n_{н} = 710 об/хв$, $I_{\text{н}} = 84 A$, $U_{\text{н}} = 220\ B$, $R_{\text{я}} + R_{\text{дп}} = 0{,}1085\ Ом$, $N = 372$, $N = 372$, $2a = 2$, $2p = 4$, $\Phi_{\text{н}} = 0{,}025 Вб$, $J_{\text{д}} = 1\ кг \cdot м^2$, $\eta = 0{,}77$, $ПВ = 40\%$

Ці двигуни відповідають необхідним техніко-експлуатаційним характеристикам, що узгоджуються з умовами експлуатації мостового крана.

2.3 Побудова та аналіз швидкісної діаграми електроприводу

Першим кроком визначимо номінальну частоту обертання електродвигуна, яка знаходиться за співвідношенням:

$$\omega_{\text{н}} = \frac{\pi n_{\text{н}}}{30},$$

$$\omega_{\text{н}} = \frac{\pi \cdot 710}{30} = 74{,}3\ c^{-1}.$$

Передавальне число редуктора можна знайти на основі кінематичних параметрів механізму пересування моста:

$$i_{\text{м}} = \frac{\omega_{\text{н}} D_{\text{м}}}{2V_{\text{м}}},$$

$$i_{\text{м}} = \frac{74{,}3 \cdot 0{,}71}{2 \cdot 1{,}25} = 27{,}7.$$

Отримане значення передавального числа використовується для подальшої побудови швидкісної та навантажувальної діаграм, що дозволяє оцінити коректність попередньо обраного типорозміру електродвигуна

Для перевірки ефективності роботи електроприводу потрібно визначити статичні моменти як при транспортуванні вантажу, так і при холостому ході.

1. Рух моста з вантажем.

Статичний момент при завантаженні становить:

$$M_{\text{ст м}} = \frac{k_p \cdot \text{g} \cdot \left(G_{\text{вп}} + m_{\text{мех}}\right)}{\cdot \eta_{\text{н м}} \cdot i_{\text{м}}} \cdot \left(\frac{\mu \cdot d_{\text{м}}}{2} + f\right),$$

$$M_{\text{ст м}} = \frac{1{,}3 \cdot 9{,}81 \cdot \left(37000 + 37000\right)}{0{,}82 \cdot 27{,}7} \cdot \left(\frac{0{,}015 \cdot 0{,}145}{2} + 0{,}5 \cdot 10^{-3}\right) = 62{,}4 Нм.$$

2. Рух моста без вантажу.

Для холостого режиму статичний момент визначається:

$$M_{\text{ст м 0}} = \frac{k_{\text{р}} \cdot m_{\text{мех}} \cdot g}{\eta_{\text{м0}} \cdot i_{\text{м}}} \left(\frac{\mu \cdot d_{м}}{2} + f\right),$$

де $\eta_{\text{м0}}$ – де ККД механізму при русі без вантажу визначається з урахуванням співвідношення:

$$\gamma = \frac{m_{\text{мΣ}}}{G_{\text{вп}} + m_{\text{мΣ}}}.$$

Після підстановки параметрів отримуємо:

$$\gamma = \frac{37000}{37000 + 37000} = 0{,}53.$$

У подальших розрахунках приймаємо $\eta_{\text{м0}} = 0{,}78$.

$$M_{\text{ст м 0}} = \frac{1{,}3 \cdot 37000 \cdot 9{,}81}{0{,}78 \cdot 27{,}7} \cdot \left(0{,}015 \cdot \frac{0{,}145}{2} + 0{,}5 \cdot 10^{-3}\right) = 37{,}7\ Нм$$

Часові параметри розгону та уповільнення обмежуються максимально допустимим прискоренням $a_{\text{м}} \le 0{,}6 м/с^2$.

Оскільки зміна швидкості по природній механічній характеристиці незначна, часи пуску $t_{\text{п м}}$ та гальмування $t_{\text{г м}}$ для різних режимів руху можна визначити:

$$t_{\text{п м}} = t_{\text{г м}} = \frac{V_{\text{м}}}{a_{\text{м}}},$$

$$t_{\text{п м}} = t_{\text{г м}} = \frac{1,25}{0,6} = 2,08\ c.$$

Час руху моста на номінальній (установленій) швидкості з ногмінальним завантаженням $t_{\text{ст м 1}}$ і час сталого руху при холостому ході $t_{\text{ст м 2}}$ визначається наступним чином:

$$t_{\text{ст м 1}} = t_{\text{ст м 2}} = \frac{L_{\text{м}} - \left(l_{\text{п м}} + l_{\text{г м}}\right)}{V_{\text{м}}},$$

де $l_{\text{п м}}$, $l_{\text{г м}}$ – шлях, пройдений при пуску і гальмуванні

$$l_{\text{п м}} = l_{\text{г м}} = \frac{V_{\text{м}}}{2} \cdot t_{\text{п м}},$$

$$l_{\text{п м}} = l_{\text{г м}} = \frac{1,25}{2} \cdot 2,08 = 1,3\ м,$$

$$t_{\text{ст м 1}} = t_{\text{ст м 2}} = \frac{45 - \left(2,08 + 2,08\right)}{1,25} = 32,8\ c.$$

2.4 Розрахунок навантажувальної діаграми електроприводу

Побудова діаграми навантаження електроприводу передбачає визначення моментів на валу двигуна на різних етапах руху моста — як у завантаженому режимі, так і без вантажу. Це дозволяє провести оцінку працездатності приводу в динамічних умовах.

Розгін завантаженого моста.

Момент, що розвивається електродвигуном під час розгону моста з вантажем, визначається так:

$$M_{\text{р м}} = M_{\text{ст м}} + M_{\text{дин}},$$

де $M_{\text{ст м}} = 62,4\ Нм$ - статичний момент при русі з вантажем (визначений у попередньому підрозділі);

$M_{\text{дин}}$ – динамічний додаток моменту, який враховує інерційність механізму:

$$M_{\text{дин}} = J_{\text{пр}} \cdot \frac{\omega_{\text{н}}}{t_{\text{п м}}},$$

$J_{\text{пр}}$ – приведений до вала електродвигуна момент інерції механізму:

$$J_{\text{пр}} = 2\delta J_{\text{дв}} + \frac{G_{\text{мех}} \cdot V_{\text{м}}^{\ 2}}{\omega_{\text{н}}^2};$$

$G_{\text{мех}}$ – сумарна маса механізму в завантаженому стані:

$$G_{\text{мех}} = G_{\text{вп}} + m_{\text{м}} + m_{\text{в}} + m_0;$$

Після підстановки числових значень отримуємо:

$$G_{\text{мех}} = 37000 + 8000 + 28000 + 1000 = 74000 кг.$$

$$J_{\text{пр}} = 2 \cdot 1,2 \cdot 1 + \frac{74000 \cdot 1,25^2}{74,3^2} = 2,4 + 18,4 = 20,8\ кг \cdot м^2,$$

$$M_{\text{дин}} = 20,8 \cdot \frac{74,3}{2,08} = 743\ Нм,$$

$$M_{\text{р м}} = 62,4 + 743 = 805,4\ Нм.$$

Рух завантаженого моста при сталій швидкості.

У режимі усталеного переміщення момент, який повинен розвивати двигун, компенсує лише статичні сили опору:

$$M_{\text{ст м 1}} = M_{\text{ст м}} = 62,4 \ Нм = 62,4 \ Нм .$$

Гальмування завантаженого моста.

Під час сповільнення дія моментів аналогічна режиму пуску, але динамічна складова змінює знак:

$$M_{\text{г м}} = M_{\text{ст м}} - M_{\text{дин}},$$

$$M_{\text{г м}} = 62,4 - 743 = -680,6 \ Нм .$$

Розгін моста без вантажу.

Для холостого режиму пусковий момент визначаємо:

$$-M_{\text{р м 0}} = M_{\text{ст м 0}} + M_{\text{дин 0}},$$

де $M_{\text{ст м 0}} = 37,7 \ Нм$ - статичний момент при русі без вантажу (визначений у попередньому розділі)

$$M_{\text{дин 0}} = J_{\text{пр 0}} \cdot \frac{\omega_{\text{н}}}{t_{\text{п м}}},$$

$J_{\text{пр 0}}$ – приведений момент інерції механізму без вантажу

$$J_{\text{пр 0}} = 2\delta J_{\text{дв}} + \frac{G_{\text{мех 0}} \cdot V_{\text{м}}^{\ 2}}{\omega_{\text{н}}^2},$$

$G_{\text{мех 0}}$ – маса механізму при русі без вантажу

$$G_{\text{мех 0}} = m_{\text{м}} + m_{\text{в}} + m_0,$$

Після підстановки числових значень отримуємо:

$$G_{\text{мех }0} = 28000 + 8000 + 1000 = 37000 кг\,,$$

$$J_{\text{пр }0} = 2 \cdot 1,2 \cdot 1 + \frac{37000 \cdot 1,25^2}{74,3^2} = 2,4 + 10,47 = 12,87\ кг \cdot м^2\,,$$

$$M_{\text{дин }0} = 12,87 \cdot \frac{74,3}{2,08} = 459,7\ Нм\,,$$

$$-M_{\text{р м }0} = 37,7 + 459,7 = 497,4\ Нм\,.$$

Рух порожнього моста на сталій швидкості.

Оскільки інерційні складові відсутні, момент у цьому режимі дорівнює:

$$-M_{\text{ст м }2} = M_{\text{ст м }0} = 37,7\ Нм\,.$$

Гальмування порожнього моста.

Уповільнення холостого ходу описується рівнянням:

$$-M_{\text{г м }0} = M_{\text{ст м }0} - M_{\text{дин }0}\,,$$

$$-M_{\text{г м }0} = 37,7 - 459,7 = -420\ Нм\,.$$

Уточнення відносної тривалості вмикання (ПВ)

На основі отриманої навантажувальної діаграми уточнюємо тривалість увімкнення електродвигуна:

$$ПВ_{\text{м}} = \frac{2 \cdot t_{\text{п м}} + 2 \cdot t_{\text{г м}} + t_{\text{ст м 1}} + t_{\text{ст м 2}}}{T}\,,$$

$$ПВ_{\text{м}} = \frac{2 \cdot 2,08 + 2 \cdot 2,08 + 2 \cdot 32,8}{480} = 0,152\,.$$

2.5 Перевірка правильності попереднього вибору потужності електродвигуна

Щоб переконатися, що вибраний для механізму пересування моста електродвигун відповідає тепловим та динамічним вимогам, виконуємо перевірку за методом еквівалентного моменту. Цей метод дозволяє оцінити вплив усіх режимів роботи протягом циклу з урахуванням нерівномірності навантаження.

Еквівалентний момент визначається виразом:

$$M_{\text{екв м}} = \sqrt{\frac{\sum M_{\text{п}_i}^2 t_{\text{п}_i} + \sum M_{\text{г}_i}^2 t_{\text{г}_i} + \sum M_{\text{ст}_i}^2 t_{\text{ст}_i}}{\beta\left(\sum t_{\text{п}_i} + \sum t_{\text{г}_i}\right) + \sum t_{\text{ст}_i}}}, \qquad (2.1)$$

де $t_{\text{п}_i}$, $t_{\text{г}_i}$, $t_{\text{ст}_i}$ – тривалість роботи механізму на кожній ділянці циклу;

$M_{\text{п}_i}$, $M_{\text{г}_i}$, $M_{\text{ст}_i}$ – середній момент на відповідній ділянці (пуск, гальмування, рух із усталеною швидкістю);

$\beta = \frac{1+\beta_0}{2} = \frac{1+0{,}35}{2} = 0{,}675$ коефіцієнт, який ураховує погіршення умов вентиляції під час пускових та гальмівних режимів;

$\beta_0 = 0{,}35$ – коефіцієнт, що враховує недостатню інтенсивність охолодження під час пауз.

Вимога до правильності вибору електродвигуна формулюється так:

$$M_{\text{н}} \geq M_{\text{екв пр}}, \qquad (2.2)$$

де

$$M_{\text{екв пр}} = \frac{M_{\text{екв м}}}{2} \cdot \sqrt{\frac{ПВ_{\text{м}}}{ПВ_{\text{ст}}}}.$$

Використавши раніше визначені моменти на різних ділянках циклу роботи та відповідні інтервали часу, підставляємо дані у формулу (2.1). У результаті отримуємо: $M_{\text{екв м}} = 112{,}8$ *Нм*

$$M_{\text{екв пр}} = \frac{112,8}{2} \cdot \sqrt{\frac{0,152}{0,4}} = 35,4 \ Нм .$$

Номінальний момент двигуна, згідно з технічними характеристиками:

$$M_{\text{н}} = \frac{P_{\text{н}}}{\omega_{\text{н}}},$$

$$M_{\text{н}} = \frac{16000}{74,3} = 215 \ Нм .$$

Оскільки виконується умова (2.2), обраний двигун забезпечує допустимий тепловий режим і не перегрівається під час циклічної роботи

Перевірка за умовами перевантажувальної здатності.

Окрім теплової оцінки, необхідно переконатися, що двигун витримує максимальний динамічний момент під час пуску.

Максимальний момент привода під час пуску становить $M_{\max} = 805,4 \ Нм$.

Оскільки привід використовує два однакових двигуни, момент на валу кожного складає: $M_{\max дв} = 805,4/2 = 402,7 \ Нм$

Перевантажувальна здатність двигунів постійного струму становить: $\lambda = 2 \div 2,5$.

Тоді максимально допустимий момент:

$$M_{\text{дв max}} = 2,5 \cdot 215 = 537,5 \ Нм$$

Оскільки виконується нерівність $M_{\text{дв max}} \geq M_{\max}$, двигун повністю задовольняє умови за допустимими перевантаженнями.

Висновок.

Виконана перевірка засвідчує, що вибраний електродвигун:

- відповідає тепловому режиму роботи за методом еквівалентного моменту;
- має достатній запас перевантажувальної здатності;

- забезпечує надійну роботу механізму моста у всіх розрахованих режимах.

За всіма критеріями вибір потужності електродвигуна є обґрунтованим і правильним.

2.6 Вибір основного електрообладнання та розрахунок параметрів силового кола електроприводу

Для живлення двох послідовно з'єднаних двигунів постійного струму необхідно забезпечити подачу випрямленої напруги, що дорівнює не менше ніж подвійній номінальній напрузі одного двигуна, а також відповідний рівень випрямленого струму, який не може бути меншим за номінальне значення струму двигуна. Виходячи з цих умов, обираємо тиристорний перетворювач типу ТПЕ-100/100-460, який має такі паспортні характеристики:

номінальна випрямлена напруга $U_{\text{dн}} = 460B$;

номінальний випрямлений струм $I_{\text{dн}} = 100A$.

Оскільки вихідна напруга перетворювача становить 460 В, його підключення до мережі 380 В, 50 Гц здійснюється через токообмежувальний реактор, який захищає силове коло від перевантажень та імпульсних кидків струму. Для цього вибираємо реактор РТСТ-8-0505-УЗ з такими параметрами: $I_{\text{р н}} = 100A$; $L_{\text{р}} = 0,505 \cdot 10^{-3} Гн$; $R_{\text{р}} = 0,032Ом$

Розрахунок електромеханічних параметрів двигуна.

Проти-ЕРС двигуна постійного струму визначається рівнянням:

$$E_{\text{дв}} = \frac{pN\omega_{\text{н}}}{2\pi a}\Phi_{\text{н}} = c\omega_{\text{н}},$$

де c – постійна проти-ЕРС

$$c = \frac{pN}{2\pi a}\Phi_{\text{н}},$$

$$c = \frac{2 \cdot 372}{2 \cdot 3{,}14 \cdot 1} \cdot 25 \cdot 10^{-3} = 2{,}8,$$

$$E_{\text{дв}} = 2{,}8 \cdot 74{,}3 = 208B.$$

Коефіцієнт підсилення еквівалентного двигуна:

$$k_{\text{д}} = \frac{1}{2 \cdot c},$$

$$k_{\text{д}} = \frac{1}{2 \cdot 2{,}8} = 0{,}179 .179.$$

Коефіцієнт підсилення тиристорнго перетворювача:

$$k_{\text{тп}} = \frac{1{,}3\ U_{\text{dн}}}{U_{\text{y max}}},$$

де $U_{\text{y max}}$ – максимальна напруга керування. Приймаємо $U_{\text{y max}} = 10\ B$

$$k_{\text{тп}} = \frac{1{,}3 \cdot 460}{10} = 59{,}8.$$

Опір силового кола електроприводу.

Активний опір якоря нагрітої машини визначається:

$$R_{\text{як}} = \alpha_t \cdot (R_{\text{я}} + R_{\text{дп}} + R_{\text{к}}),$$

де $R_{\text{я}}$, $R_{\text{дп}}$, $R_{\text{к}}$ – опір обмотки якоря, опір обмотки додаткових полюсів і опір компенсаційної обмотки;

α_t – коефіцієнт, що враховує збільшення опору при нагріві

$$R_{\text{як}} = 1{,}36 \cdot 0{,}1085 = 0{,}1477 Ом.$$

Сумарний опір силового кола:

$$R_{\Sigma} = 2R_{як} + 2R_{р},$$

$$R_{\Sigma} = 2 \cdot 0{,}1477 + 2 \cdot 0{,}032 = 0{,}36 \ Ом.$$

Індуктивність силового кола.

Загальна індуктивність:

$$L_{\Sigma} = 2 \cdot L_{як} + 2 \cdot L_{р},$$

де $L_{як}$ – індуктивність обмотки якоря;

$$L_{як} = k_1 \frac{U_н}{I_н \omega_н p},$$

$k = 0{,}6$ – для некомпенсованих машин.

$$L_{як} = 0{,}6 \frac{220}{84 \cdot 74{,}3 \cdot 2} = 0{,}0105 \ Гн,$$

Після розрахунку:

$$L_{\Sigma} = 2 \cdot 0{,}0105 + 2 \cdot 0{,}505 \cdot 10^{-3} = 0{,}022 Гн.$$

Постійні часу електроприводу.

Електромагнітна постійна часу силового кола

$$T_е = \frac{L_{\Sigma}}{R_{\Sigma}},$$

$$T_е = \frac{0{,}022}{0{,}36} = 0{,}061 \ с.$$

Електромагнітна постійна часу якоря двигуна

$$T_a = \frac{L_{як}}{R_{як}},$$

$$T_a = \frac{0,0105}{0,1477} = 0,071\ c.$$

Електромеханічна постійна часу

$$T_м = J_{пр} \cdot \frac{R_\Sigma}{(2 \cdot c)^2},$$

$$T_м = 20,8 \cdot \frac{0,36}{(2 \cdot 2,8)^2} = 0,24\ c.$$

2.7 Вибір структури системи автоматичного управління

Якість перехідних процесів у системі електроприводу визначається співвідношенням постійних часу окремих її елементів. Для отримання бажаних динамічних характеристик застосовують корегуючі регулятори, які дозволяють компенсувати інерційність об'єкта керування та забезпечити стійкість і точність системи.

Побудова системи автоматичного регулювання включає:

1. складання структурної схеми електроприводу;
2. формування передатних функцій окремих ланок;
3. уведення регуляторів у ті контури, де необхідна компенсація інерцій чи покращення якості регулювання.

У системах електроприводів найпоширенішим підходом є підлегле (каскадне) регулювання, де для кожного регульованого параметра створюється власний контур. Кожен контур містить два елементи: об'єкт регулювання та відповідний регулятор.

Обрана структура системи автоматичного керування.

У даному електроприводі застосовується двоконтурна система з такими характеристиками:

- внутрішній контур — контур регулювання струму;

- зовнішній контур — регулювання за ЕРС (аналог швидкісного контуру).

Для спрощення конструкції датчик ЕРС не використовується. Замість цього здійснюється підсумовування сигналів, пропорційних напрузі та падінню напруги на якорі, безпосередньо на вході регулятора ЕРС.

Системи з регулюванням швидкості за ЕРС застосовуються у випадках, коли:

- точність стабілізації швидкості не є критично високою;
- необхідні простота та надійність конструкції;
- широке використання у загальнопромислових електроприводах.

Така структура повністю відповідає вимогам до привода механізму пересування моста та забезпечує оптимальне поєднання економічності, стабільності та невибагливості в експлуатації.

2.8 Синтез послідовного коригувального елемента для контуру регулювання струму

Оскільки контур струму у підлеглих системах автоматичного регулювання зазвичай виконує роль внутрішнього контуру, саме його параметри визначають динаміку роботи всієї багатоконтурної САР. Тому початковий етап налагодження та розрахунку системи традиційно розпочинають саме з цього контуру. Коректний підбір параметрів у контурі струму є ключовою умовою забезпечення необхідної швидкодії електроприводу, оскільки саме тут закладаються основні характеристики реакції електромеханічної системи на зовнішні збурення та команди керування.

Об’єктом регулювання в цьому контурі є сукупність елементів, які включають якірне коло двигуна, силовий тиристорний перетворювач разом із системою імпульсно-фазового керування (СІФУ) та датчик струму.

Якір двигуна у динамічному відношенні моделюється аперіодичною ланкою, що характеризується коефіцієнтом підсилення $1/R_{\Sigma}$ та електромагнітною постійною часу T_{e}.

Динамічні властивості тиристорного перетворювача разом із СІФУ описуються передавальною функцією:

$$W_{\text{тп}}(p) = \frac{k_{\text{тп}}}{1 + \tau p} \ ,$$

де $k_{\text{тп}}$ – коефіцієнт підсилення перетворювача;

τ – середньостатистичне запізнення, зумовлене побудовою і режимом роботи тиристорної схеми.

Середнє запізнення визначають за співвідношенням

$$\tau = \frac{1}{2mf} .$$

де m – число пульсацій для трифазного мостового випрямляча, m=6;

f – частота мережі живлення.

$$\tau = \frac{1}{2 \cdot 6 \cdot 50} = 0,00167 \ c^{-1} .$$

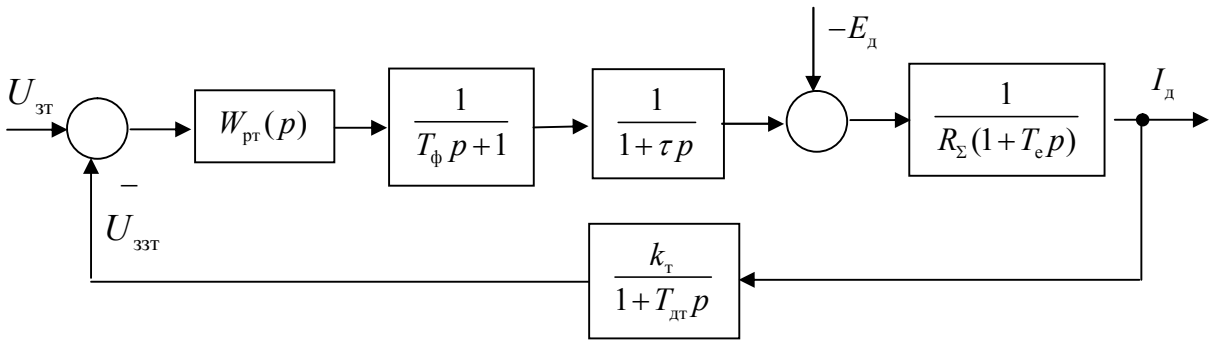
На виході датчика струму розташований фільтр для згладжування пульсацій, що також можна описати аперіодичною ланкою з постійною часу $T_{\text{дт}}$. Її значення визначають таким чином, щоб рівень пульсацій не порушував стабільність роботи СІФУ та не спричиняв збоїв в аналогових підсилювачах.

Крім того, на вході СІФУ встановлюють ще один фільтр з постійною часу $T_{\text{ф}}$, необхідний для коректної обробки сигналу керування.

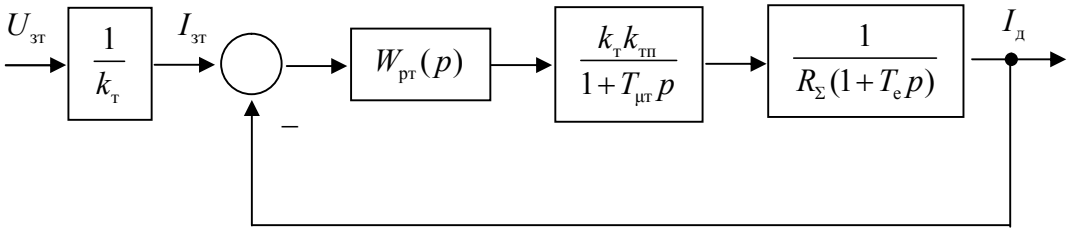
Алгоритмічна структура контуру подана на рис. 2.1,а.

Перехід до спрощеної алгоритмічної моделі.

З метою застосування стандартних методик синтезу послідовних коригувальних пристроїв (ПКП), схему доцільно перетворити до виду системи з одиничним зворотним зв’язком. Така модифікована структура подана на рис.2.1,б.



а)



б)

Рисунок 2.1 –Алгоритмічні структури контуру регулювання струму:
а – вихідна (базова) структурна схема;
б – схема після перетворення до вигляду
з одиничним зворотним зв’язком.

Дві аперіодичні ланки з малими постійними часу $T_{дт}$, $T_ф$ та τ (які істотно менші за електромагнітну постійну часу двигуна T_e) об’єднують в одну, отримуючи зведену «малу» некомпенсовану постійну часу:

$$T_{мт} = \tau + T_{дт} + T_ф .$$

Беручи до уваги, що ці значення досить незначні, приймемо, що $T_{дт} = 0,5 \cdot 10^{-3} c$; $T_ф = 1 \cdot 10^{-3} c$. тоді:

$$T_{мт} = 1,67 \cdot 10^{-3} + 0,5 \cdot 10^{-3} + 1 \cdot 10^{-3} = 3,17 \cdot 10^{-3}\ c .$$

Форсувальний елемент $(1 + T_{дт} p)$, який розташований у структурі перед входом замкненого контуру з одиничним зворотним зв’язком, під час визна-

чення малої некомпенсованої сталої часу контуру швидкості, відповідно до класичної методики синтезу регуляторів, у розрахунок не включають.

Ц дозволяє суттєво спростити структуру та перейти до розрахункової схеми, наведеної на рисунку 2.2.

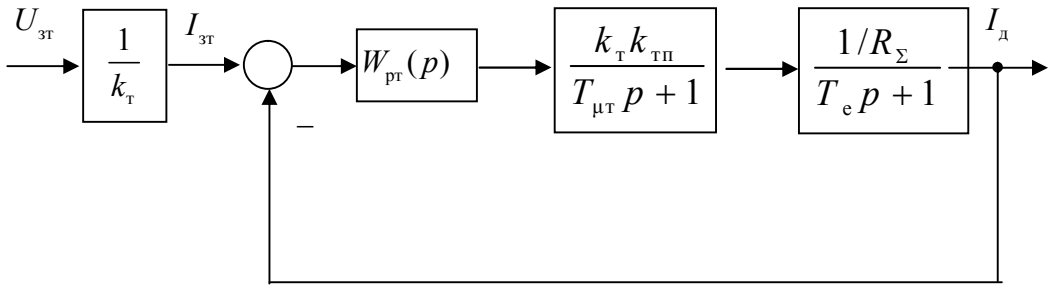


Рисунок 2.2 – Узагальнена розрахункова алгоритмічна модель контуру струму.

Синтез регулятора струму.

Контур регулювання струму оптимізують за модульним критерієм. Оптимізований контур в розімкненому стані повинен мати передавальну функцію:

$$W_{бт}(p) = \frac{1}{2T_{μт} p(1 + T_{μт} p)} \ .$$

Передавальна функція об’єкта, згідно зі схемою на рис. 2.2:

$$W_{об}(p) = \frac{k_{об}}{(1 + T_е p)(1 + T_{μт} p)} \ ,$$

де $k_{об} = k_т k_{тп} R_{яΣ}^{-1}$ – коефіцієнт підсилення об’єкта.

Тоді передавальна функція регулятора струму має вигляд:

$$W_{рт}(p) = \frac{W_{бт}(p)}{W_{об}(p)} = k_{рт} \cdot \frac{1 + T_е p}{p} \ ,$$

де $k_{рт} = \frac{R_Σ}{2T_{μт} k_{тр} k_т}$ – коефіцієнт підсилення регулятора.

Отриману передавальну функцію регулятора струму $W_{\text{рт}}(p)$ можна реалізувати у вигляді стандартного пропорційно-інтегрального (ПІ) регулятора.

Визначення коефіцієнта зворотного зв'язку.

Коефіцієнт посилення каналу зворотного зв'язку по струму $k_{\text{т}}$ визначаємо з

$$I_{\text{д max}} \cdot k_{\text{т}} = U_{\text{зт max}} ,$$

$I_{\text{д max}}$ – максимально допустимий струм якоря приймаємо рівним $I_{\text{ш}} = I_{\text{д max}} = 2I_{\text{н}} = 2 \cdot 84 = 168A$;

$U_{\text{зт max}}$ – максимальна напруга завдання, приймемо $U_{\text{зт max}} = 24\ B$

$$k_{\text{т}} = \frac{U_{\text{зт max}}}{I_{\text{д max}}},$$

Звідси отримаємо коефіцієнт підсилення зворотного зв'язку по струму:

$$k_{\text{т}} = \frac{24}{168} = 0{,}143\ B / A .$$

У результаті отримуємо повну передавальну функцію оптимізованого за модульним критерієм контуру регулювання струму:

$$W_{\text{зт}}(p) = \frac{1/k_{\text{т}}}{2T_{\mu\text{т}}^2 p^2 + 2T_{\mu\text{т}} p + 1} .$$

2.9 Синтез послідовного коригувального пристрою контуру ЕРС

Алгоритмічна структура системи з включеним зворотним зв'язком за ЕРС наведена на рисунку 2.3.

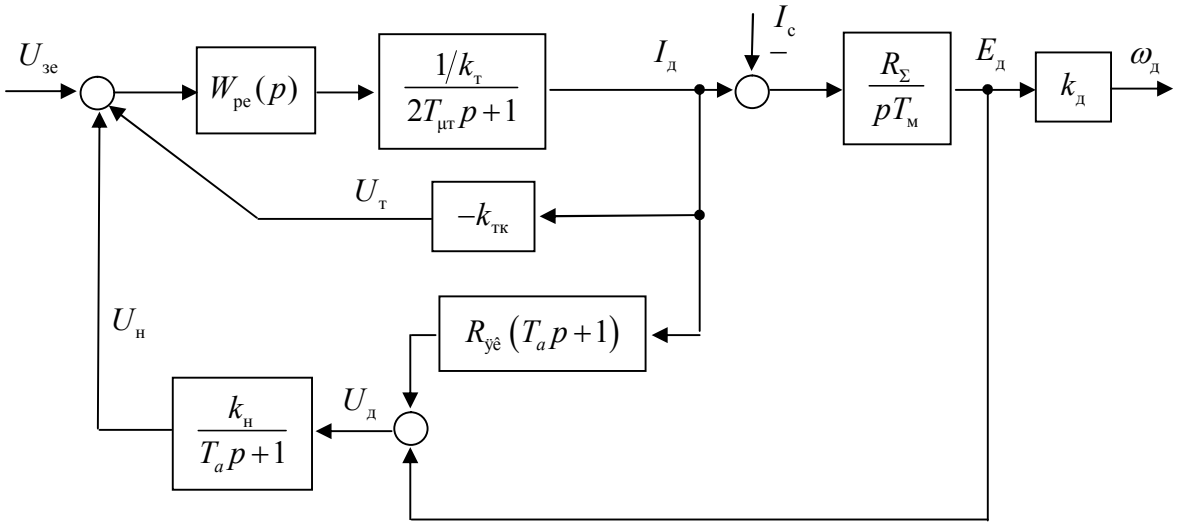


Рисунок 2.3 – Алгоритмічна структура системи регулювання із зворотним зв’язком за ЕРС

Контур струму, який був попередньо оптимізований відповідно до модульного критерію, описується передавальною функцією:

$$W_т(p)=\frac{1/k_т}{2T_{цт}p+1};$$

(доданком $2T_{цт}^2p^2$ у знаменнику нехтуємо, оскільки він є дуже малим).

Коефіцієнт підсилення ланцюга струмової компенсації приймається рівним

$$k_{тк}=k_н R_{як}\ , \qquad (2.3)$$

де $k_н$ – коефіцієнт підсилення ланцюга зворотного зв’язку по напрузі.

В цьому випадку алгоритмічну схему на рис. 2.3 можна перетворити до вигляду, показаного на рис. 2.4.

При обраному значенні $k_{тк}$ на вхід регулятора ЕРС надходить сигнал, пропорційний проти-ЕРС:

$$U_е=U_н-U_т=\frac{k_н}{T_a p+1}E_д\ .$$

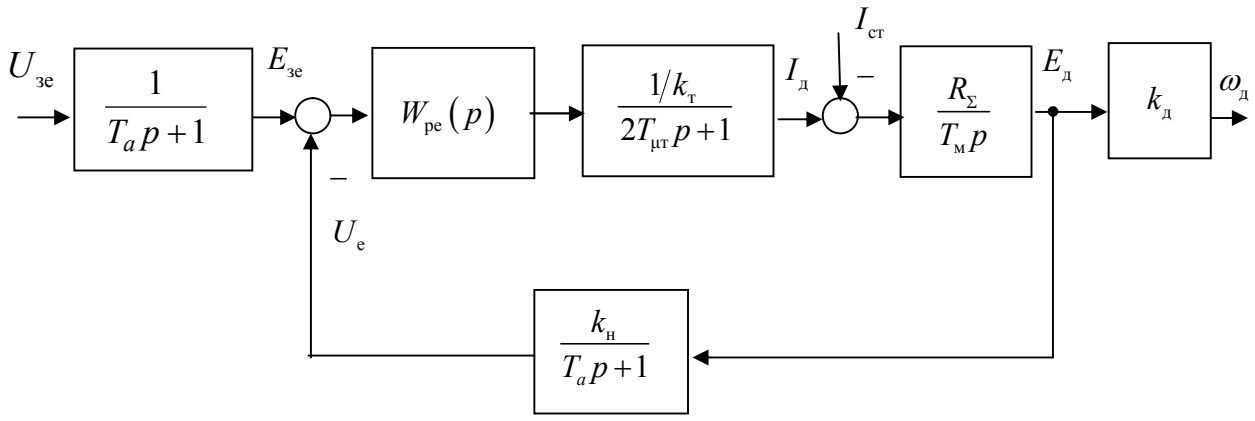


Рисунок 2.4 – Модифікована алгоритмічна схема системи

У зв’язку з тим. що в ланцюзі зворотного зв’язку за напругою присутня аперіодична ланка з передавальною функцією

$$\frac{k_н}{T_a p+1},$$

де T_a – – електромагнітна постійна часу якірного кола двигуна, то для запобігання підвищеному перерегулюванню струму при збуренні по задаючій дії на вхід системи вводиться додаткова аперіодична ланка з аналогічною передавальною функцією. Розрахункова схема такої системи наведена на рис. 2.5.

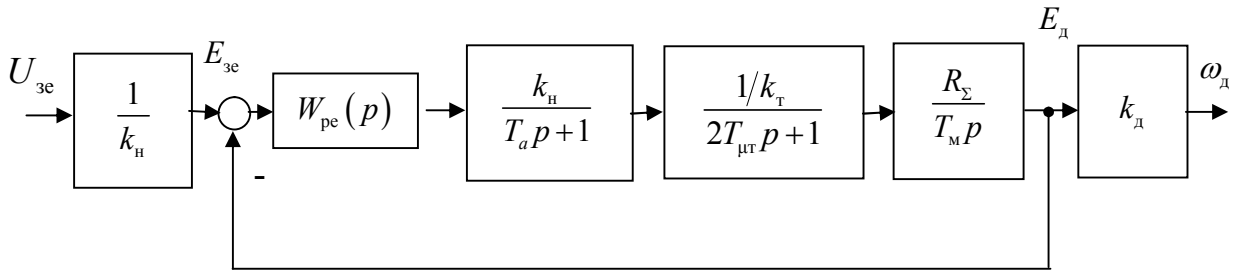


Рисунок 2.5 – Розрахункова алгоритмічна модель системи з одиничним зворотним зв’язком

При оптимізації контуру ЕРС за симетричним критерієм застосовується ПІ-регулятор, передавальна функція якого описується виразом:

$$W_{\text{ре}}(p) = k_{\text{ре}} \frac{4T_{\text{це}}p + 1}{4T_{\text{це}}p} \ ,$$

де $k_{\text{ре}} = \dfrac{T_{\text{м}} k_{\text{т}}}{2k_{\text{н}} T_{\text{це}} R_{\Sigma}}$ – коефіцієнт підсилення регулятора ЕРС

$T_{\text{це}}$ – мала некомпенсуєма постійна часу контура ЕРС

$$T_{\text{це}} = 2T_{\text{цт}} + T_a ,$$

$$T_{\text{це}} = 0,071 + 2 \cdot 0,00317 = 0,077 \ c .$$

Коефіцієнт підсилення ланцюга зворотного зв'язку за напругою визначається як:

$$k_{\text{н}} = \frac{U_{\text{зе max}}}{E_{\text{дв max}}} \ ,$$

де $E_{\text{д max}}$ – максимальне значення ЕРС еквівалентного двигуна.

$U_{\text{зе max}}$ – відповідна напруга завдання.

Приймаємо конкретні значення $U_{\text{зе max}} = 24 \ B$ та $E_{\text{д max}} = 2U_{\text{н}} = 440B$.

$$k_{\text{н}} = \frac{24}{440} = 0,05454 .$$

Визначаємо коефіцієнт підсилення ланцюга струмової компенсації згідно з формулою (2.3).

$$k_{\text{тк}} = 0,05454 \cdot 0,1477 = 8,05 \cdot 10^{-3} .$$

На рисунку 2.6 представлена повна алгоритмічна схема системи з урахуванням обох контурів: струмового та ЕРС.

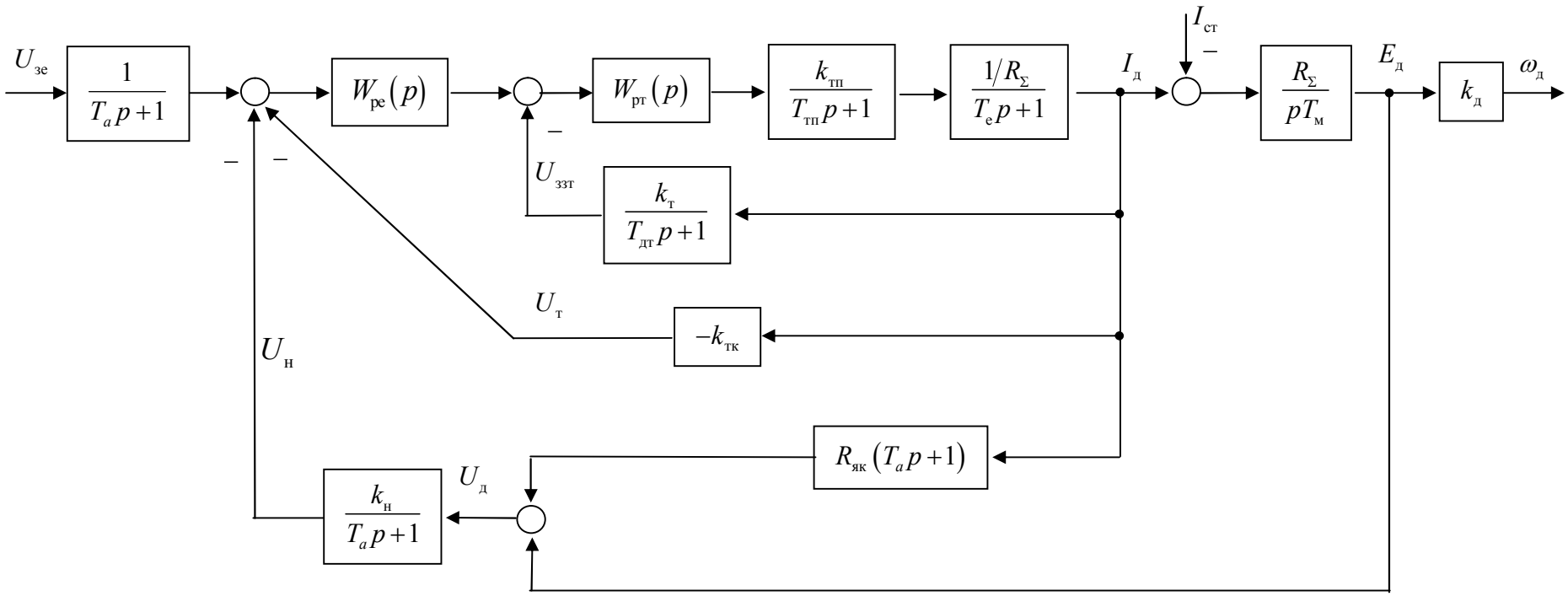


Рисунок 2.6 – Узагальнена алгоритмічна схема системи після синтезу регуляторів струму та ЕРС

Передавальні функції системи, оптимізованої за симетричним критерієм, мають вигляд:

$$W_{\omega_д U_{зе}}(p) = \frac{\omega_д(p)}{U_{зе}(p)} = \frac{k_д}{k_н} \cdot \frac{1 + 4T_{μе}p}{8T^3_{μе}p^3 + 8T^2_{μе}p^2 + 4T_{μе}p + 1},$$

$$W_{\omega_д I_{ст}}(p) = \frac{\omega_д(p)}{I_с(p)} = \frac{8T^2_{μе}k_д R_Σ}{T_м} \cdot \frac{p(1 + T_{μе}p)}{8T^3_{μе}p^3 + 8T^2_{μе}p^2 + 4T_{μе}p + 1},$$

$$W_{I_д U_{зе}}(p) = \frac{I_д(p)}{U_{зе}(p)} = \frac{T_м}{k_н R_Σ} \cdot \frac{p(1 + 4T_{μе}p)}{8T^3_{μе}p^3 + 8T^2_{μе}p^2 + 4T_{μе}p + 1},$$

$$W_{I_д I_{ст}}(p) = \frac{I_д(p)}{I_{ст}(p)} = \frac{1 + 4T_{μе}p}{8T^3_{μе}p^3 + 8T^2_{μе}p^2 + 4T_{μе}p + 1}.$$

Далі алгоритмічну схему системи підлеглого регулювання швидкості з контуром струму, оптимізованим по модульному критерію, та контуром ЕРС, налаштованим по симетричному критерію, перетворюють до вигляду, показаного на рис. 2.7.

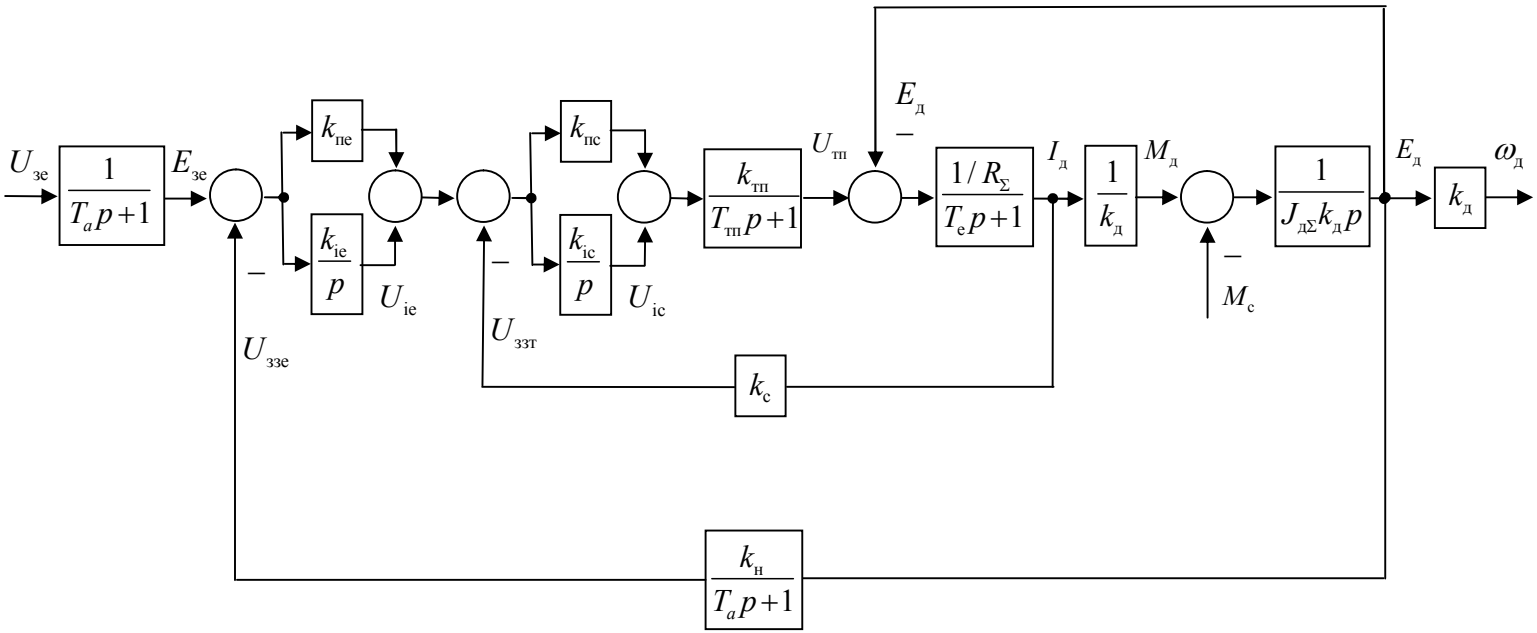


Рисунок 2.7 – Перетворена структурно-алгоритмічна схема системи регулювання швидкості

Дане перетворення виконується за умови, що коефіцієнт підсилення ланцюга струмової компенсації вибраний рівним $k_{тк} = k_н R_{як}$.

На схемі рис. 2.7 прийняті такі позначення:

U_{ie} – напруга на виході інтегральної частини регулятора ЕРС;

$k_{пе}$, k_{ie} – коефіцієнти підсилення пропорційної та інтегральної частин регулятора ЕРС.

Значення параметрів $k_{пе}$ та k_{ie} можуть бути визначені безпосередньо з передавальної функції регулятора ЕРС, що дозволяє встановити необхідні коефіцієнти підсилення для забезпечення заданих динамічних характеристик системи.

$$W_{ре}(p) = \frac{T_м k_с}{2T_{μе} k_н R_Σ} \cdot \frac{4T_{μе} p + 1}{4T_{μе} p} = \frac{T_м k_т}{2T_{μе} k_н R_Σ} + \frac{T_м k_с}{8T_{μе}^2 k_н R_Σ} \frac{1}{p} = k_{пе} + \frac{k_{ie}}{p},$$

$$k_{пе} = \frac{T_м k_с}{2T_{μе} k_н R_Σ}, \quad k_{ie} = \frac{T_м k_с}{8T_{μе}^2 k_н R_Σ}.$$

U_{ic} – напруга на виході інтегральної частини регулятора струму;

$k_{пе}$, k_{ie} – коефіцієнти підсилення пропорційної і інтегральної частин регулятора струму.

Значення параметрів $k_{пе}$ та k_{ie} визначаються на основі передавальної функції регулятора струму, що дозволяє забезпечити потрібну динаміку та точність регулювання струму в системі.

$$W_{рс}(p) = \frac{R_Σ}{2T_{μс} k_{тп} k_с} \cdot \frac{1 + T_е p}{p} = \frac{R_Σ T_е}{2T_{μс} k_{тп} k_с} + \frac{R_Σ}{2T_{μс} k_{тп} k_с} \frac{1}{p} = k_{пс} + \frac{k_{ic}}{p},$$

$$k_{пс} = \frac{R_Σ T_е}{2T_{μт} k_{тп} k_с}, \quad k_{ic} = \frac{R_Σ}{2T_{μс} k_{тп} k_с}.$$

При формуванні передавальної функції двигуна замість $T_м$ використано відповідний вираз

$$T_{\text{м}} = \frac{R_{\Sigma} \cdot J_{\text{д}\Sigma}}{c_{\text{ек}}^2} = R_{\Sigma} J_{\text{д}\Sigma} k_{\text{д}}^2; \quad k_{\text{д}} = 1 / c_{\text{ек}}.$$

На схемі показано зворотний зв'язок по ЕРС самого двигуна, що забезпечує корекцію впливу його електромеханічних властивостей на роботу системи.

Алгоритмічну схему системи можна представити у спрощеному вигляді, як показано на рис. 2.8.

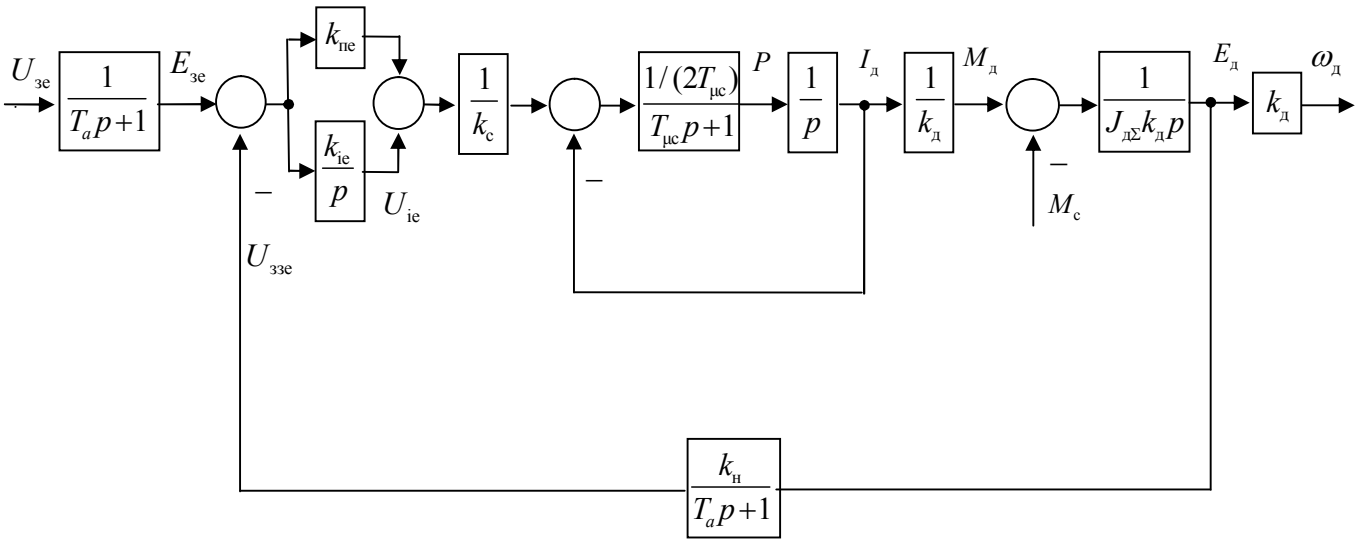


Рисунок 2.8 – Спрощена узагальнена алгоритмічна схема системи регулювання

На схемі контур струму, оптимізований за модульним критерієм, зображено як послідовне з'єднання аперіодичної та інтегруючої ланок, охоплених одиничним зворотним зв'язком. Сигнал на виході першої ланки пропорційний ривку електроприводу P, що дозволяє оцінювати динамічну реакцію системи на зміни задаючого впливу.

2.10 Аналіз динамічних характеристик системи

Дослідження перехідних процесів системи було проведено з використанням програмного комплексу MATLAB, що забезпечує широкий набір засобів для моделювання динаміки електроприводів. Цей пакет дозволяє виконувати розрахунки перехідних процесів як по передавальних функціях системи, так і за допомогою векторно-матричних рівнянь. Крім того, у середовищі SIMULINK можна створювати схематичні моделі системи, які відображають реальні структурні взаємозв'язки між регуляторами, виконавчими механізмами та датчиками.

Для моделювання розглянутої системи була побудована детальна модель, представлена на рисунку 2.9. У цій моделі враховані всі основні елементи алгоритмічної схеми системи, представленої на рис. 2.7, включаючи контур струму, регулятор ЕРС та ланцюги зворотного зв'язку. Для спрощеного аналізу динаміки також була створена компактна модель, наведена на рис. 2.10, яка відповідає спрощеній алгоритмічній схемі системи з рис. 2.8 і дозволяє ефективно оцінювати перехідні процеси без втрати загальної динамічної картини.

На рисунках 2.11 та 2.12 представлені графіки динамічної реакції системи на вплив задаючої та збурюючої дії. Графіки відображають зміну основних змінних стану: швидкості $\omega_{\text{д}}$ та струму $I_{\text{д}}$ двигуна протягом перехідного процесу.

При первинному налаштуванні системи на симетричний оптимум було виявлено значне перерегулювання швидкості на рівні 43,3%. Така ситуація пояснюється високою швидкодією контурів і відсутністю додаткової фільтрації задаючого сигналу. Для зниження перерегулювання було рекомендовано включити вхідний фільтр з постійною часу $4T_{\mu\text{e}}$. Застосування цього фільтра дозволяє зменшити перерегулювання до 8,1%, проте збільшується час досягнення першого узгодження системи, що є типовим компромісом між швидкодією та стабільністю перехідного процесу. На графіках перехідних процесів на рисун-

ках 2.11 та 2.12 криві $\omega_д = f(t)$ і $I_д = f(t)$, отримані з використанням фільтра та без нього, наведені окремо для наочного порівняння і позначені 1 і 2 відповідно.

У результаті проведеного моделювання встановлено, що оптимізація контуру струму за модульним критерієм і контуру ЕРС за симетричним критерієм забезпечує бажану точність регулювання. Використання додаткового фільтра дозволяє ефективно керувати перерегулюванням і підвищує стійкість системи до збурень, що особливо важливо для промислових електроприводів, де надійність і передбачуваність перехідних процесів мають критичне значення.

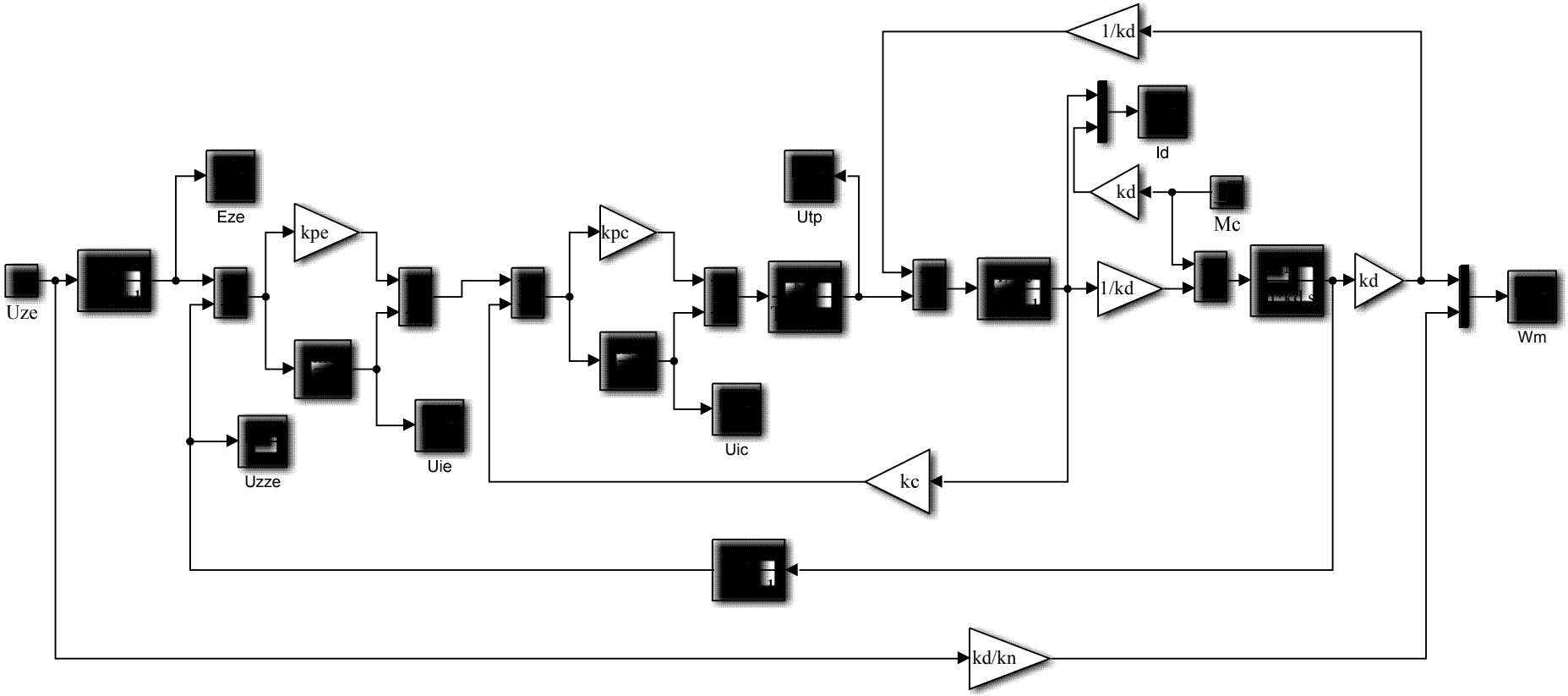


Рисунок 2.9 – Детальна модель системи, розроблена в SIMULINK/MATLAB на основі алгоритмічної схеми рис. 2.7

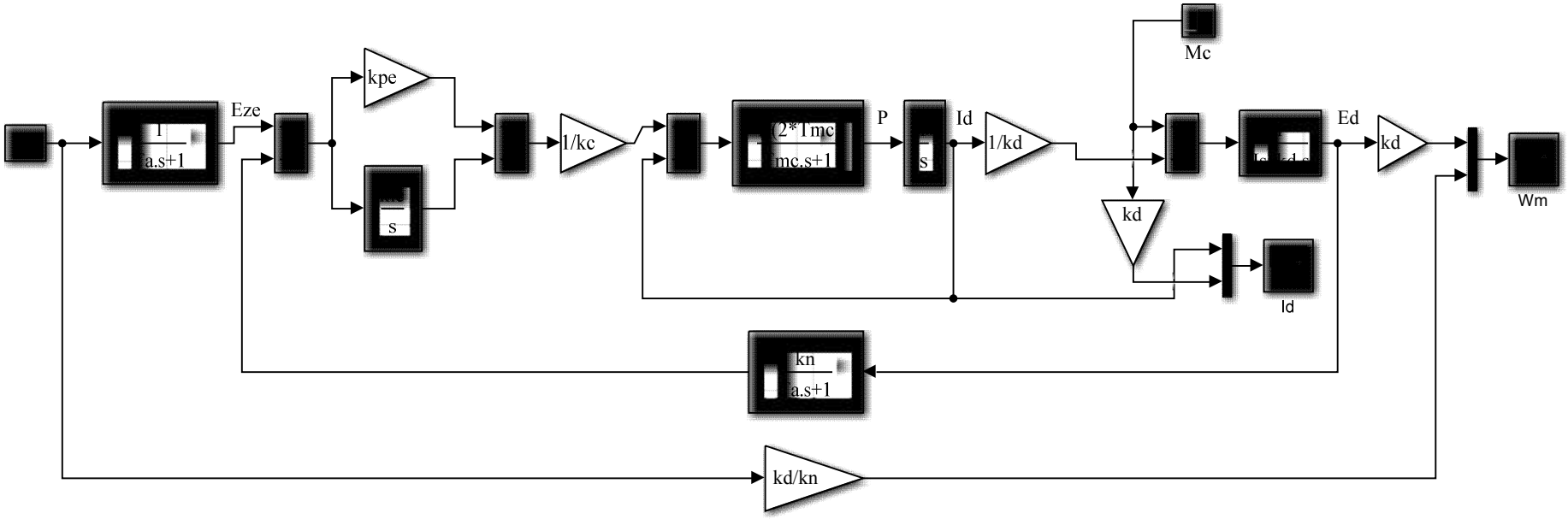
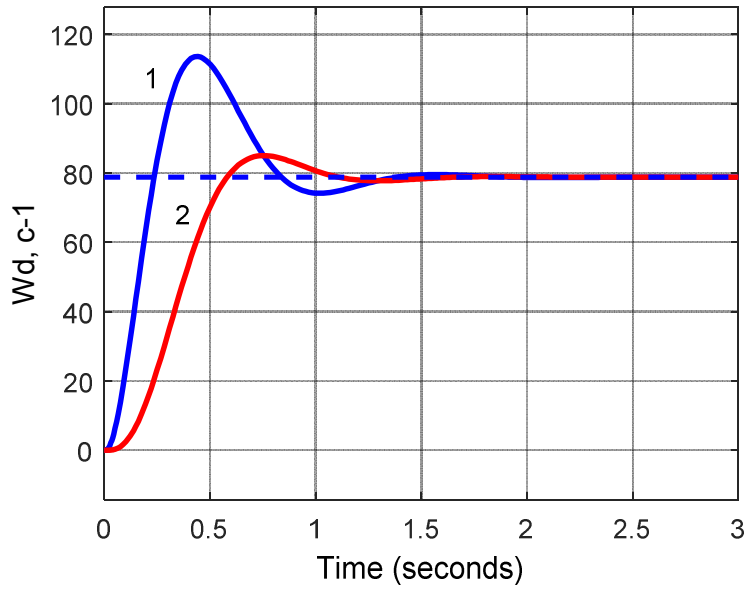
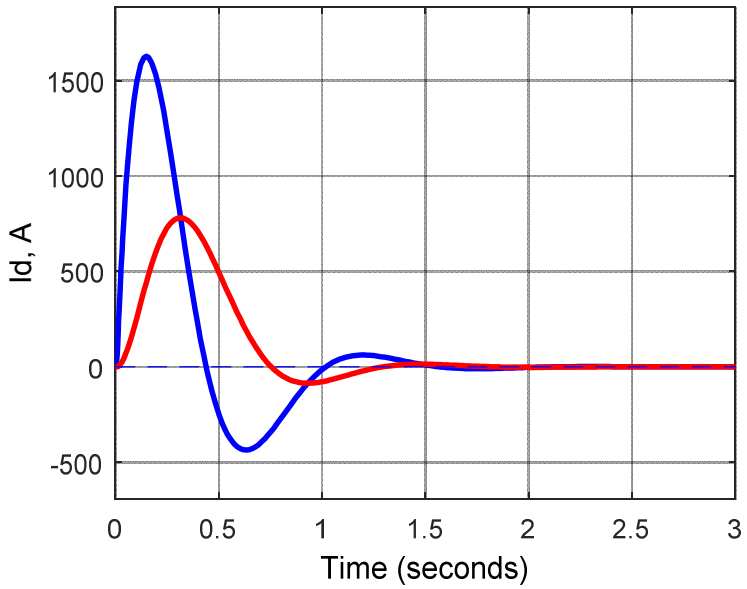


Рисунок 2.10 – Спрощена модель системи в SIMULINK/MATLAB,
створена на основі алгоритмічної схеми рис. 2.8



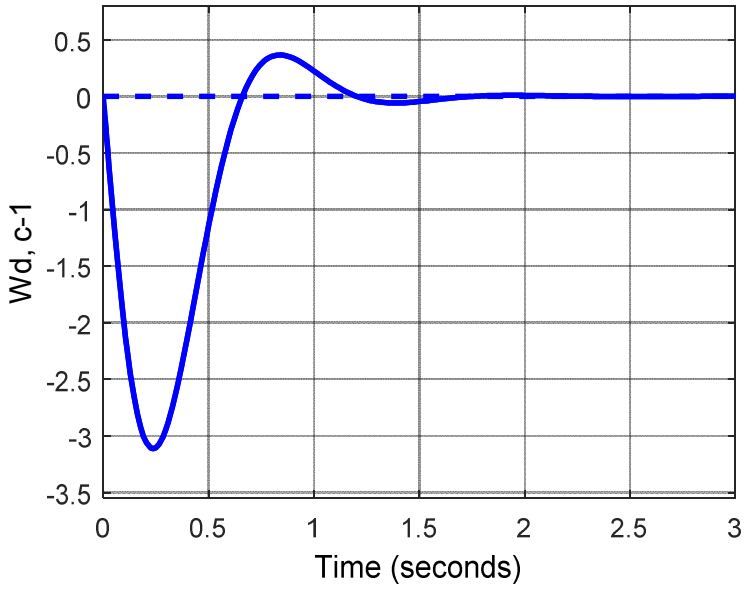
а)



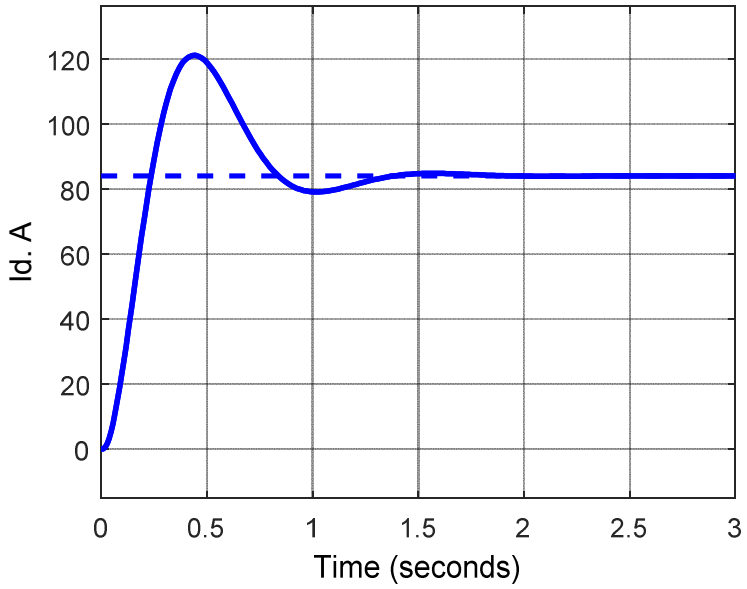
б)

Рисунок 2.11 – Динаміка змінних стану системи під впливом задаючої дії:

а) – швидкість $\omega_д = f(t)$; б) – струм $I_д = f(t)$



а)



б)

Рисунок 2.12 – Динаміка змінних стану системи під впливом збурюючої дії:

а) – швидкість $\omega_д = f(t)$; б) – струм $I_д = f(t)$

ВИСНОВКИ ДО РОЗДІЛУ 2

1. На основі аналізу технічних вимог до електроприводів основних механізмів мостових кранів для механізму переміщення моста обрано двохдвигунний електропривод постійного струму, виконаний за схемою ТП-Д. Для забезпечення точного управління координатами електроприводу застосована система підлеглого регулювання, яка дозволяє ефективно контролювати як струм, так і швидкість двигунів.

2. Проведено вибір електродвигунів і основного силового обладнання електроприводу, а також виконано перевірку їх роботи щодо нагріву та перевантажувальної здатності. Для цього розраховано швидкісні та навантажувальні діаграми електроприводу. Отримані результати еквівалентного моменту показали, що обрані двигуни відповідають експлуатаційним вимогам щодо нагріву, а їх перевантажувальна здатність також забезпечує надійну роботу в умовах експлуатаційних перевантажень.

3. Виконано розрахунок параметрів силового кола електроприводу, які необхідні для подальшого проектування та аналізу системи автоматичного управління, що забезпечує коректну роботу електродвигунів у різних режимах.

4. Обрана структура системи автоматичного управління дозволяє забезпечити необхідні механічні характеристики електроприводу як у статичних, так і в динамічних режимах, включаючи високу якість перехідних процесів та оптимізацію роботи електроприводу під час різних режимів роботи механізму. Для механізму переміщення моста було обрано двоконтурну систему підлеглого регулювання: внутрішній контур контролює струм, а зовнішній – швидкість, при цьому система замкнена по ЕРС електродвигуна, що забезпечує стабільність та точність регулювання.

5. Виконано синтез послідовних коригуючих пристроїв для окремих контурів системи. Оскільки характер перехідних процесів визначається співвідношенням постійних часу елементів системи, для підвищення якості переходових процесів застосовувалися регулятори, які коригують ці співвідношення. Для

контурів струму та ЕРС вибрано пропорційно-інтегральні (ПІ) регулятори. Це дозволило оптимізувати контур струму за модульним критерієм, а контур ЕРС – за симетричним критерієм, забезпечуючи стабільність і точність регулювання.

6. Проведено комп'ютерне моделювання розробленої системи з використанням пакету MATLAB та середовища SIMULINK. Було створено детальну схему моделі системи, яка дозволяє досліджувати перехідні процеси як під впливом задаючої дії, так і при збуреннях. Аналіз проводився також за допомогою векторно-матричних рівнянь і передавальних функцій системи. Під час налаштування системи на симетричний оптимум спостерігалося значне перерегулювання швидкості, що склало 43,3%. Для його зменшення доцільно було додати вхідний фільтр з постійною часу $4T_{\mu e}$, що дозволило знизити перерегулювання до 8,1%, при цьому трохи збільшився час першого узгодження системи, що є типовим компромісом між швидкодією та стабільністю перехідного процесу.

РОЗДІЛ 3
ДОСЛІДЖЕННЯ ХАРАКТЕРИСТИК СИСТЕМИ З УРАХУВАННЯМ ПРУЖНИХ ЕЛЕМЕНТІВ

3.1 Розрахункові схеми механічної частини електроприводу

При аналізі динамічних характеристик електроприводу механізму переміщення моста зазвичай розглядають механічні зв'язки між рухомими масами як абсолютно жорсткі. У такому випадку механічна частина представлена у вигляді єдиної приведеної ланки, що дає середнє уявлення про рух елементів, але не враховує вплив пружності реальних механічних з'єднань.

У реальній системі через обмежену жорсткість зв'язків механічна частина електроприводу поводиться як пружна система. Будь-який додаток моменту двигуна або обурююча дія, наприклад, момент від статичного навантаження, викликає коливання пов'язаних мас. Ці коливання підвищують максимальні навантаження на елементи зв'язків і впливають на точність виконання заданих переміщень та швидкостей.

Пружність механізму переміщення моста обумовлена як кінцевою жорсткістю з'єднання між двигуном і мостом, так і коливаннями вантажу. У загальному випадку система може розглядатися як трьохмасова. Проте, оскільки жорсткість редукторів та інших передач значно перевищує жорсткість, що визначає коливання вантажу, останні при аналізі навантажень можна не враховувати. У таких умовах відхилення вантажу приймаються практично постійними під час розгону, і систему можна зводити до двохмасової моделі.

Для механізму пересування моста застосовується багатодвигунний електропривод із двома двигунами. У загальному вигляді така конфігурація зводиться до розгалуженої розрахункової схеми, наведеної на рис. 3.1.

На схемі позначено: $J_{д1}$, $J_{д2}$ – моменти інерції відповідних двигунів; c_1, c_2 – приведені жорсткості кожного приводу; $J_м$ – приведений момент інерції механізму (приведений до валу двигуна момент інерції моста).

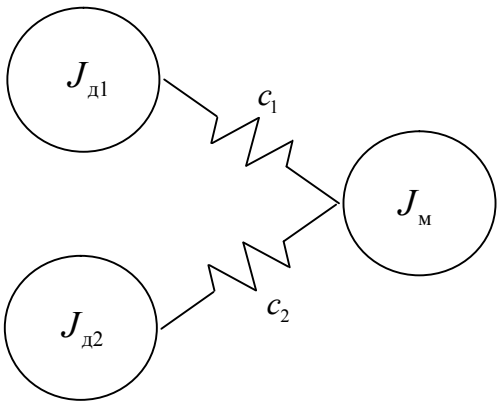


Рисунок 3.1 – Розгалужена розрахункова схема
механічної частини електроприводу

З урахуванням того. що обидва двигуни рухаються синфазно, розгалужену схему можна спростити і представити у вигляді двомасової розрахункової моделі, показаної на рис. 3.2.

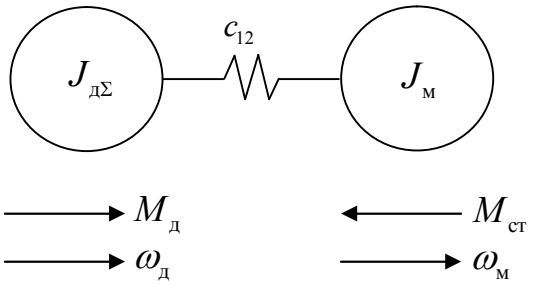


Рисунок 3.2 – Двомасова розрахункова схема
механічної частини електроприводу

У цій схемі: $J_{дΣ}$ – сумарний момент двигуна разом із жорстко пов'язаними з ним елементами механічної частини; c_{12} – приведена жорсткість зв'язку між масами

Параметри схеми визначаються наступними співвідношеннями.

$$J_{дΣ} = 1,2(J_{д1} + J_{д2}) = 1,2 \cdot 2 \cdot J_д,$$

$$J_{дΣ} = 1,2 \cdot 2 \cdot 1 = 2,4 \ кг \cdot м^2.$$

$$c_{12} = c_1 + c_2 .$$

Приведений момент інерції моста до валу двигуна визначається як:

$$J_{\text{м}} = \frac{G_{\text{мех}} \cdot V_{\text{м}}^{\ 2}}{\omega_{\text{н}}^2},$$

$$J_{\text{м}} = \frac{66000 \cdot 1,25^2}{74,3^2} = 18,4 \ кг \cdot м^2 .$$

3.2 Рівняння стану одномасової системи

Щоб сформувати математичну модель двомасової системи, спершу необхідно описати рівняння стану одномасової системи. Такі рівняння відображають динаміку системи і мають наступний загальний вигляд:

$$\begin{cases} \dfrac{d\omega_{\text{д}}}{dt} = \dfrac{2k\Phi_{\text{н}}}{J_{\text{дΣ}}} I_{\text{д}} - \dfrac{1}{J_{\text{дΣ}}} M_{\text{ст}}; \\ \dfrac{dI_{\text{д}}}{dt} = -\dfrac{2k\Phi_{\text{н}}}{R_{\Sigma} T_{\text{е}}} \omega_{\text{д}} - \dfrac{1}{T_{\text{е}}} I_{\text{д}} + \dfrac{1}{R_{\Sigma} T_{\text{е}}} U_{\text{тп}}; \\ \dfrac{dU_{\text{тп}}}{dt} = -\dfrac{k_{\text{с}} k_{\text{пс}} k_{\text{тп}}}{T_{\mu\text{с}}} I_{\text{д}} - \dfrac{1}{T_{\mu\text{с}}} U_{\text{тп}} - \\ \qquad + \dfrac{k_{\text{тп}}}{T_{\mu\text{с}}} U_{\text{іс}} + \dfrac{k_{\text{тп}} k_{\text{пс}}}{T_{\mu\text{с}}} U_{\text{іе}} + \dfrac{k_{\text{пе}} k_{\text{пс}} k_{\text{тп}}}{T_{\mu\text{с}}} E_{\text{зе}} - \dfrac{k_{\text{пе}} k_{\text{пс}} k_{\text{тп}}}{T_{\mu\text{с}}} U_{\text{ззе}}; \\ \dfrac{dU_{\text{іс}}}{dt} = -k_{\text{іс}} k_{\text{с}} I_{\text{д}} + k_{\text{іс}} U_{\text{іе}} + k_{\text{пе}} k_{\text{іс}} E_{\text{зе}} - k_{\text{пе}} k_{\text{іс}} U_{\text{ззе}}; \\ \dfrac{dU_{\text{іе}}}{dt} = k_{\text{іе}} E_{\text{зе}} - k_{\text{іе}} U_{\text{ззе}}; \\ \dfrac{dE_{\text{зе}}}{dt} = -\dfrac{1}{T_a} E_{\text{зе}} + \dfrac{1}{T_a} U_{\text{зе}}; \\ \dfrac{dU_{\text{ззе}}}{dt} = \dfrac{k_{\text{н}}}{k_{\text{д}} T_a} \omega_{\text{д}} - \dfrac{1}{T_a} U_{\text{ззе}}. \end{cases} \tag{3.1}$$

Для зручності подальших розрахунків систему можна записати у векторно-матричній формі, доповнивши її рівнянням виходу. Введемо вектор стану системи $\vec{X}(t)$ розмірності n та вектор управління $\vec{U}(t)$ розмірності m:

$$\vec{X}(t)=\left[\omega_{\text{д}}(t),I_{\text{д}}(t),U_{\text{тп}}(t),U_{\text{ic}}(t),U_{\text{ie}}(t),E_{\text{зе}}(t),U_{\text{ззе}}(t)\right]^T ,$$

$$\vec{U}(t)=\left[U_{\text{зе}}(t),M_{\text{ст}}(t)\right]^T ,$$

де T – означає операцію транспонування.

У цьому випадку рівняння стану та вихід системи можна подати у стандартній матричній формі

$$\begin{cases}\dot{\vec{X}}(t)=\mathbf{A}\vec{X}(t)+\mathbf{B}\vec{U}(t);\\ \vec{Y}(t)=\mathbf{C}\vec{X}(t),\end{cases}$$

де **A** – матриця стану системи розмірності $n\times n$; **B** – матриця впливу керування розмірності $n\times m$; **C** - матриця виходу розміру $r\times n$; $\vec{Y}(t)$ - вектор виходу розміру r.

Матриці **A** і **B** одномасової системи:

$$\mathbf{A}=\begin{array}{c}\begin{matrix}\omega_{\text{д}} & I_{\text{д}} & U_{\text{тп}} & U_{\text{ic}} & U_{\text{ie}} & E_{\text{зе}} & U_{\text{ззе}}\end{matrix}\\ \begin{vmatrix}0 & \frac{2k\Phi_{\text{н}}}{J_{\text{д}\Sigma}} & 0 & 0 & 0 & 0 & 0\\ -\frac{2k\Phi_{\text{н}}}{R_{\Sigma}T_{\text{e}}} & -\frac{1}{T_{\text{e}}} & \frac{1}{R_{\Sigma}T_{\text{e}}} & 0 & 0 & 0 & 0\\ 0 & -\frac{k_{\text{c}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{c}}} & -\frac{1}{T_{\mu\text{c}}} & \frac{k_{\text{тп}}}{T_{\mu\text{c}}} & \frac{k_{\text{тп}}k_{\text{пс}}}{T_{\mu\text{c}}} & \frac{k_{\text{пе}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{c}}} & -\frac{k_{\text{пе}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{c}}}\\ 0 & -k_{\text{ic}}k & 0 & 0 & k_{\text{ic}} & k_{\text{пе}}k_{\text{ic}} & -k_{\text{пе}}k_{\text{ic}}\\ 0 & 0 & 0 & 0 & 0 & k_{\text{ie}} & -k_{\text{ie}}\\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_a} & 0\\ \frac{k_{\text{н}}}{k_{\text{д}}T_a} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_a}\end{vmatrix}\end{array}$$

$$\mathbf{B}=\begin{array}{c}\begin{matrix}U_{\text{зе}} & M_{\text{ст}}\end{matrix}\\ \begin{vmatrix}0 & -\frac{1}{J_{\text{д}\Sigma}}\\ 0 & 0\\ 0 & 0\\ 0 & 0\\ 0 & 0\\ \frac{1}{T_a} & 0\\ 0 & 0\end{vmatrix}\end{array}$$

У випадку одномасової системи, оптимізований контур, настроєний за модульним критерієм, може бути представлений як послідовне з'єднання аперіодичної та інтегруючої ланок із одиничним зворотним зв'язком. Сигнал на виході аперіодичної ланки пропорційний прискоренню електроприводу P.

Рівняння стану цієї спрощеної одномасової системи набуває вигляду:

$$\begin{cases} \dfrac{d\omega_{\text{д}}}{dt} = \dfrac{1}{J_{\text{д}\Sigma}} M_{\text{д}} - \dfrac{1}{J_{\text{д}\Sigma}} M_{\text{ст}}; \\ \dfrac{dM_{\text{д}}}{dt} = \dfrac{1}{k_{\text{д}}} P; \\ \dfrac{dP}{dt} = -\dfrac{k_{\text{д}}}{2T_{\mu\text{c}}^2} M_{\text{д}} - \dfrac{1}{T_{\mu\text{c}}} P + \dfrac{k_{\text{ре}}}{2T_{\mu\text{c}}^2 k_{\text{с}}} E_{\text{зе}} - \dfrac{k_{\text{ре}}}{2T_{\mu\text{c}}^2 k_{\text{с}}} U_{\text{ззе}} + \dfrac{1}{2T_{\mu\text{c}}^2 k_{\text{с}}} U_{\text{ie}}; \\ \dfrac{dU_{\text{ie}}}{dt} = k_{\text{ie}} E_{\text{зе}} - k_{\text{ie}} U_{\text{ззе}}; \\ \dfrac{dE_{\text{зе}}}{dt} = -\dfrac{1}{T_a} E_{\text{зе}} + \dfrac{1}{T_a} U_{\text{зе}}; \\ \dfrac{dU_{\text{ззе}}}{dt} = \dfrac{k_{\text{н}}}{T_a k_{\text{д}}} \omega_{\text{д}} - \dfrac{1}{T_a} U_{\text{ззе}}. \end{cases} \tag{3.2}$$

Тут $M_{\text{д}}$ - момент еквівалентного двигуна: $M_{\text{д}} = 2k\Phi_{\text{н}} I_{\text{д}}$

Складові рівнянь стану спрощеної одномасової системи у векторно-матричній формй визначаються наступним чином:

$$\overrightarrow{X}(t) = \left[\omega_{\text{д}}(t), M_{\text{д}}(t),\ P(t),\ E_{\text{зе}}(t),\ U_{\text{ззе}}(t),\ U_{\text{ie}}(t) \right]^T$$

$$\overrightarrow{U}(t) = \left[U_{\text{зе}}(t), M_{\text{ст}}(t) \right]^T ,$$

Матриці **A** і **B** спрощеної системи:

$$\begin{matrix} & \omega_{\text{д}} & M_{\text{д}} & P & E_{\text{зе}} & U_{\text{ззе}} & U_{\text{іе}} \end{matrix}$$

$$\mathbf{A}=\begin{vmatrix} 0 & \frac{1}{J_{\text{дΣ}}} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{k_{\text{д}}} & 0 & 0 & 0 \\ 0 & -\frac{k_{\text{д}}}{2T_{\text{μс}}^2} & -\frac{1}{T_{\text{μс}}} & \frac{k_{\text{ре}}}{2T_{\text{μс}}^2 k_{\text{с}}} & -\frac{k_{\text{ре}}}{2T_{\text{μс}}^2 k_{\text{с}}} & \frac{1}{2T_{\text{μс}}^2 k_{\text{с}}} \\ 0 & 0 & 0 & -\frac{1}{T_a} & \frac{1}{T_a} & 0 \\ \frac{k_{\text{н}}}{T_a k_{\text{д}}} & 0 & 0 & 0 & -\frac{1}{T_a} & 0 \\ 0 & 0 & 0 & k_{\text{іе}} & -k_{\text{іе}} & 0 \end{vmatrix} \qquad \begin{matrix} & U_{\text{зе}} & M_{\text{ст}} \end{matrix}$$

$$\mathbf{B}=\begin{vmatrix} 0 & -\frac{1}{J_{\text{дΣ}}} \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{T_a} & 0 \\ 0 & 0 \\ 0 & 0 \end{vmatrix}$$

3.3 Математична модель двомасової системи

Кінематична схема механічної частини двомасової системи електроприводу моста з урахуванням пружних елементів і внутрішнього в'язкого тертя описується рівняннями:

$$\begin{cases} J_{\text{м}} \dfrac{d\omega_{\text{м}}}{dt} = M_{\Sigma} - M_{\text{ст}}; \\ \dfrac{dM_{\text{пр}}}{dt} = c_{12}(\omega_{\text{д}} - \omega_{\text{м}}); \\ J_{\text{дΣ}} \dfrac{d\omega_{\text{д}}}{dt} = M_{\text{д}} - M_{\Sigma}. \end{cases} \tag{3.3}$$

де M_{Σ} – сумарний момент, що передається пружною передачею, дорівнює сумі пружного моменту $M_{\text{пр}}$ і моменту внутрішнього в'язкого тертя $M_{\text{вт}} = \beta_{12}(\omega_{\text{д}} - \omega_{\text{м}})$:

$$M_{\Sigma} = M_{\text{пр}} + \beta_{12}(\omega_{\text{д}} - \omega_{\text{м}}), \tag{3.4}$$

β_{12} – коефіцієнт внутрішнього в'язкого тертя, який у розрахунках приймається рівним $\beta_{12} = 0$.

Система (3.3), з урахуванням (3.4), може бути записана у формі рівнянь Коші:

$$\begin{cases} \dfrac{d\omega_{\text{м}}}{dt} = \dfrac{1}{J_{\text{м}}} M_{\text{пр}} + \dfrac{\beta_{12}}{J_{\text{м}}} \omega_{\text{д}} - \dfrac{\beta_{12}}{J_{\text{м}}} \omega_{\text{м}} - \dfrac{1}{J_{\text{м}}} M_{\text{ст}}; \\ \dfrac{dM_{\text{пр}}}{dt} = c_{12}\omega_{\text{д}} - c_{12}\omega_{\text{м}}; \\ \dfrac{d\omega_{\text{д}}}{dt} = \dfrac{1}{J_{\text{д}\Sigma}} M_{\text{д}} - \dfrac{1}{J_{\text{д}\Sigma}} M_{\text{пр}} - \dfrac{\beta_{12}}{J_{\text{д}\Sigma}} \omega_{\text{д}} + \dfrac{\beta_{12}}{J_{\text{д}\Sigma}} \omega_{\text{м}}. \end{cases} \tag{3.5}$$

Для двомасової системи підлеглого регулювання швидкості, у якій контур струму налаштований за модульним критерієм, а контур ЕРС – за симетричним, алгоритмічна схема подана на рис. 3.3. Аналогічно одномасовій системі, її можна перетворити у спрощену схему, показану на рис. 3.4.

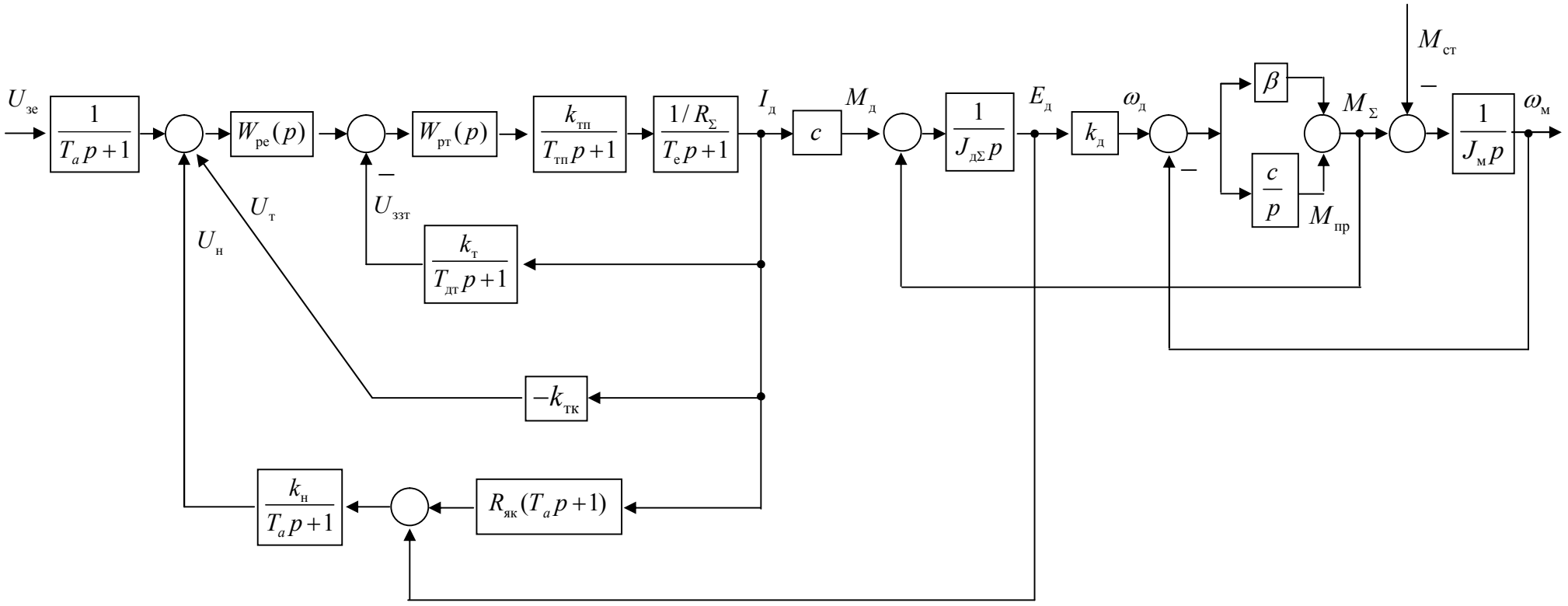


Рисунок 3.3 – Алгоритмічна схема двомасової системи

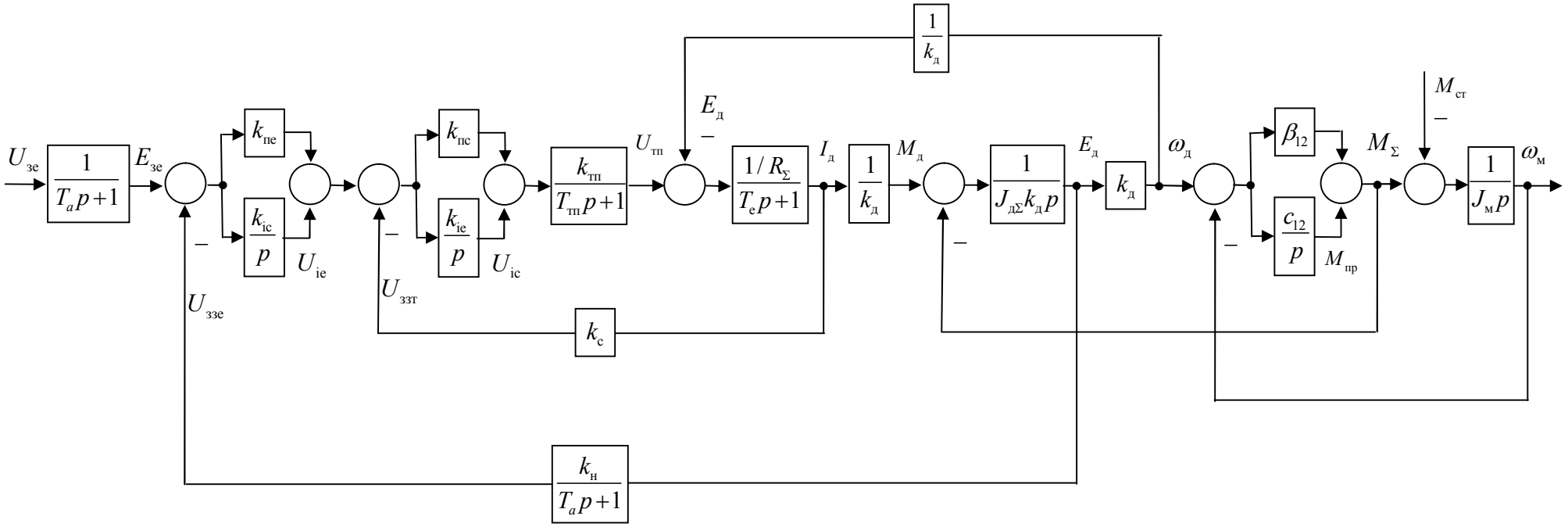


Рисунок 3.4 – Перетворена алгоритмічна схема двомасової системи

Об’єднавши рівняння одномасової (3.1) та двомасової (3.5) систем, отримуємо повну систему рівнянь стану двомасової моделі системи підлеглого регулювання швидкості з зворотним зв’язком по ЕРС двигуна:

$$
\begin{cases}
\dfrac{d\omega_{\text{м}}}{dt} = -\dfrac{\beta_{12}}{J_{\text{м}}}\omega_{\text{м}} + \dfrac{1}{J_{\text{м}}}M_{\text{пр}} + \dfrac{\beta_{12}}{J_{\text{м}}}\omega_{\text{д}} - \dfrac{1}{J_{\text{м}}}M_{\text{ст}}; \\
\dfrac{dM_{\text{пр}}}{dt} = c_{12}\omega_{\text{д}} - c_{12}\omega_{\text{м}}; \\
\dfrac{d\omega_{\text{д}}}{dt} = \dfrac{2k\Phi_{\text{н}}}{J_{\text{д}\Sigma}}I_{\text{д}} - \dfrac{1}{J_{\text{д}\Sigma}}M_{\text{пр}} - \dfrac{\beta_{12}}{J_{\text{д}\Sigma}}\omega_{\text{д}} + \dfrac{\beta_{12}}{J_{\text{д}\Sigma}}\omega_{\text{м}}; \\
\dfrac{dI_{\text{д}}}{dt} = -\dfrac{2k\Phi_{\text{н}}}{R_{\Sigma}T_{\text{е}}}\omega_{\text{д}} - \dfrac{1}{T_{\text{е}}}I_{\text{д}} + \dfrac{1}{R_{\Sigma}T_{\text{е}}}U_{\text{тп}}; \\
\dfrac{dU_{\text{тп}}}{dt} = -\dfrac{k_{\text{с}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{с}}}I_{\text{д}} - \dfrac{1}{T_{\mu\text{с}}}U_{\text{тп}} - \\
\qquad + \dfrac{k_{\text{тп}}}{T_{\mu\text{с}}}U_{\text{іс}} + \dfrac{k_{\text{тп}}k_{\text{пс}}}{T_{\mu\text{с}}}U_{\text{іе}} + \dfrac{k_{\text{пе}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{с}}}E_{\text{зе}} - \dfrac{k_{\text{пе}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{с}}}U_{\text{ззе}}; \\
\dfrac{dU_{\text{іс}}}{dt} = -k_{\text{іс}}k_{\text{с}}I_{\text{д}} + k_{\text{іс}}U_{\text{іе}} + k_{\text{пе}}k_{\text{іс}}E_{\text{зе}} - k_{\text{пе}}k_{\text{іс}}U_{\text{ззе}}; \\
\dfrac{dU_{\text{іе}}}{dt} = k_{\text{іе}}E_{\text{зе}} - k_{\text{іе}}U_{\text{ззе}}; \\
\dfrac{dE_{\text{зе}}}{dt} = -\dfrac{1}{T_a}E_{\text{зе}} + \dfrac{1}{T_a}U_{\text{зе}}; \\
\dfrac{dU_{\text{ззе}}}{dt} = \dfrac{k_{\text{н}}}{k_{\text{д}}T_a}\omega_{\text{д}} - \dfrac{1}{T_a}U_{\text{ззе}}.
\end{cases}
\tag{3.6}
$$

Складові рівнянь стану двомасової системи у векторно-матричній формі набувають вигляду:

$$\vec{X}(t) = \left[\omega_{\text{м}}(t), M_{\text{пр}}(t), \omega_{\text{д}}(t), I_{\text{д}}(t), U_{\text{тп}}(t), U_{\text{іс}}(t), U_{\text{іе}}(t), E_{\text{зе}}(t), U_{\text{ззе}}(t)\right]^T,$$

$$\vec{U}(t) = \left\{U_{\text{зе}}(t),\ M_{\text{ст}}(t)\ \right\}^T,$$

Структура матриць **A** і **B** двомасової системи

$$\mathbf{A}=\begin{vmatrix}
\omega_{\text{м}} & M_{\text{пр}} & \omega_{\text{д}} & I_{\text{д}} & U_{\text{тп}} & U_{\text{іс}} & U_{\text{іе}} & E_{\text{зе}} & U_{\text{ззе}} \\
-\dfrac{\beta_{12}}{J_{\text{м}}} & \dfrac{1}{J_{\text{м}}} & \dfrac{\beta_{12}}{J_{\text{м}}} & 0 & 0 & 0 & 0 & 0 & 0 \\
-c_{12} & 0 & c_{12} & 0 & 0 & 0 & 0 & 0 & 0 \\
\dfrac{\beta_{12}}{J_{\text{д}\Sigma}} & -\dfrac{1}{J_{\text{д}\Sigma}} & -\dfrac{\beta_{12}}{J_{\text{д}\Sigma}} & \dfrac{2k\Phi_{\text{н}}}{J_{\text{д}\Sigma}} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\dfrac{2k\Phi_{\text{н}}}{R_{\Sigma}T_{\text{е}}} & -\dfrac{1}{T_{\text{е}}} & \dfrac{1}{R_{\Sigma}T_{\text{е}}} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -\dfrac{k_{\text{с}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{с}}} & -\dfrac{1}{T_{\mu\text{с}}} & \dfrac{k_{\text{тп}}}{T_{\mu\text{с}}} & \dfrac{k_{\text{тп}}k_{\text{пс}}}{T_{\mu\text{с}}} & \dfrac{k_{\text{пе}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{с}}} & -\dfrac{k_{\text{пе}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{с}}} \\
0 & 0 & 0 & -k_{\text{іт}}k_{\text{т}} & 0 & 0 & k_{\text{іт}} & k_{\text{пе}}k_{\text{іт}} & -k_{\text{пе}}k_{\text{іт}} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & k_{\text{іе}} & -k_{\text{іе}} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -\dfrac{1}{T_a} & 0 \\
0 & 0 & \dfrac{k_{\text{н}}}{k_{\text{д}}T_a} & 0 & 0 & 0 & 0 & 0 & -\dfrac{1}{T_a}
\end{vmatrix}$$

$$\mathbf{B}=\begin{vmatrix}
U_{\text{зе}} & M_{\text{ст}} \\
0 & -\dfrac{1}{J_{\text{м}}} \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
\dfrac{1}{T_a} & 0 \\
0 & 0
\end{vmatrix}$$

Подібно до алгоритмічної схеми одномасової системи, алгоритмічна схема двомасової системи може бути представлена у спрощеному вигляді, яке наведено на рис. 3.5**.**

У цьому випадку рівняння стану двомасової системи набувають наступного вигляду:

$$\begin{cases} \dfrac{d\omega_{\text{м}}}{dt} = \dfrac{1}{J_{\text{м}}} M_{\text{пр}} + \dfrac{\beta_{12}}{J_{\text{м}}} \omega_{\text{д}} - \dfrac{\beta_{12}}{J_{\text{м}}} \omega_{\text{м}} - \dfrac{1}{J_{\text{м}}} M_{\text{ст}}; \\ \dfrac{dM_{\text{пр}}}{dt} = c_{12}\omega_{\text{д}} - c_{12}\omega_{\text{м}}; \\ \dfrac{d\omega_{\text{д}}}{dt} = \dfrac{1}{J_{\text{д}\Sigma}} M_{\text{д}} - \dfrac{1}{J_{\text{д}\Sigma}} M_{\text{пр}} - \dfrac{\beta}{J_{\text{д}\Sigma}} \omega_{\text{д}} - \dfrac{\beta}{J_{\text{д}\Sigma}} \omega_{\text{м}}; \\ \dfrac{dM_{\text{д}}}{dt} = \dfrac{1}{k_{\text{д}}} P; \\ \dfrac{dP}{dt} = -\dfrac{k_{\text{д}}}{2T_{\mu\text{c}}^2} M_{\text{д}} - \dfrac{1}{T_{\mu\text{c}}} P + \dfrac{1}{2T_{\mu\text{c}}^2 k_{\text{c}}} U_{\text{ie}} + \dfrac{k_{\text{ре}}}{2T_{\mu\text{c}}^2 k_{\text{c}}} E_{\text{зе}} - \dfrac{k_{\text{ре}}}{2T_{\mu\text{c}}^2 k_{\text{c}}} U_{\text{ззе}}; \\ \dfrac{dU_{\text{ie}}}{dt} = k_{\text{ie}} E_{\text{зе}} - k_{\text{ie}} U_{\text{ззе}}; \\ \dfrac{dE_{\text{зе}}}{dt} = -\dfrac{1}{T_a} E_{\text{зе}} + \dfrac{1}{T_a} U_{\text{зе}}; \\ \dfrac{dU_{\text{ззе}}}{dt} = \dfrac{k_{\text{н}}}{T_a k_{\text{д}}} \omega_{\text{д}} - \dfrac{1}{T_a} U_{\text{ззе}}. \end{cases} \tag{3.7}$$

Складові рівняннь стану спрощеної двомасової системи у векторно-матричній формі набувають вигляду:

$$\overrightarrow{X}(t) = \left[\omega_{\text{м}}(t),\ M_{\text{пр}}(t),\ \omega_{\text{д}}(t), M_{\text{д}}(t),\ P(t),\ U_{\text{ie}}(t),\ E_{\text{зе}}(t),\ U_{\text{ззе}}(t) \right]^T,$$

$$\overrightarrow{U}(t) = \left\{ U_{\text{зе}}(t),\ M_{\text{ст}}(t)\ \right\}^T,$$

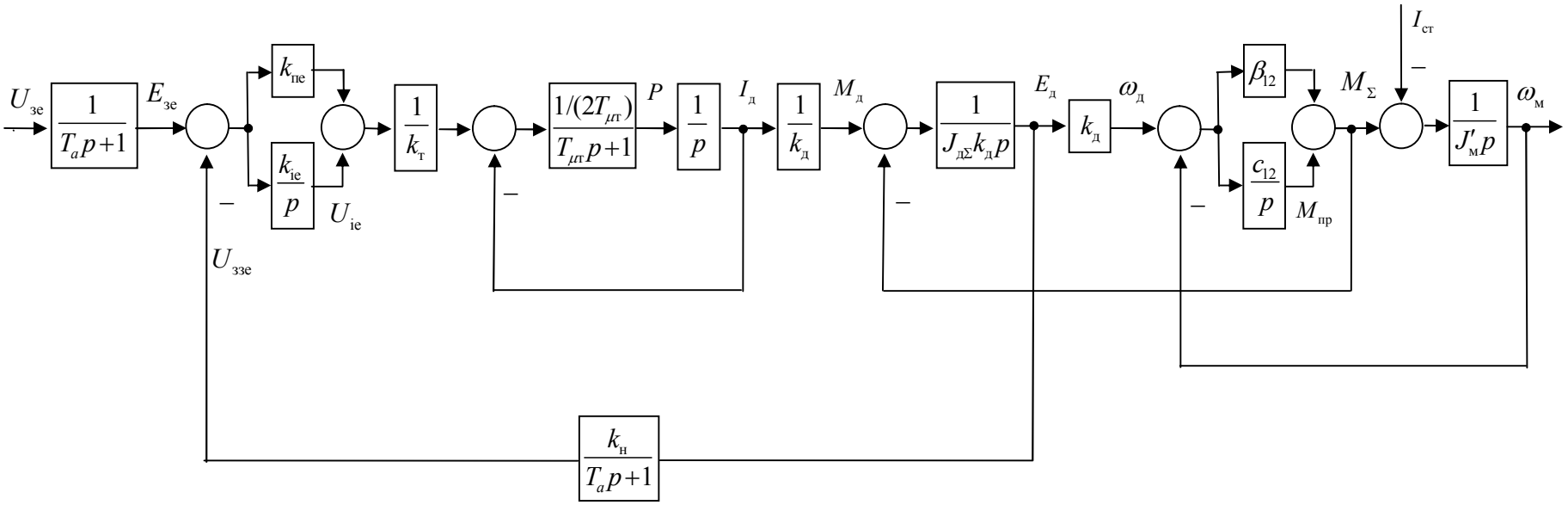


Рисунок 3.5 – Спрощена алгоритмічна схема двомасової системи

Матриці **A** і **B** спрощеної двомасової системи:

$$\mathbf{A} = \begin{array}{c} \begin{matrix} \omega_{\text{м}} & M_{\text{пр}} & \omega_{\text{д}} & M_{\text{д}} & P & U_{\text{ie}} & E_{\text{зе}} & U_{\text{ззе}} \end{matrix} \\ \begin{vmatrix} -\dfrac{\beta}{J_{\text{м}}} & \dfrac{1}{J_{\text{м}}} & \dfrac{\beta}{J_{\text{м}}} & 0 & 0 & 0 & 0 & 0 \\ -c & & c & 0 & 0 & 0 & 0 & 0 \\ \dfrac{\beta}{J_{\text{д}}} & -\dfrac{1}{J_{\text{д}}} & -\dfrac{\beta}{J_{\text{д}}} & \dfrac{1}{J_{\text{д}}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \dfrac{1}{k_{\text{д}}} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\dfrac{k_{\text{д}}}{2T_{\mu\text{c}}^2} & -\dfrac{1}{T_{\mu\text{c}}} & \dfrac{1}{2T_{\mu\text{c}}^2 k_{\text{c}}} & \dfrac{k_{\text{ре}}}{2T_{\mu\text{т}}^2 k_{\text{c}}} & -\dfrac{k_{\text{ре}}}{2T_{\mu\text{c}}^2 k_{\text{c}}} \\ 0 & 0 & 0 & 0 & 0 & 0 & k_{\text{ie}} & -k_{\text{ie}} \\ 0 & 0 & 0 & 0 & 0 & 0 & -\dfrac{1}{T_a} & 0 \\ 0 & 0 & \dfrac{k_{\text{н}}}{T_a k_{\text{д}}} & 0 & 0 & 0 & 0 & -\dfrac{1}{T_a} \end{vmatrix} \end{array} \quad \mathbf{B} = \begin{array}{c} \begin{matrix} U_{\text{зе}} & M_{\text{ст}} \end{matrix} \\ \begin{vmatrix} 0 & -\dfrac{1}{J_{\text{м}}} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \dfrac{1}{T_a} & 0 \\ 0 & 0 \\ 0 & 0 \end{vmatrix} \end{array}$$

3.4 Моделювання перехідних процесів двомасової системи

Для дослідження динаміки двомасової системи електроприводу переміщення моста було виконано розрахунок перехідних процесів із використанням пакету прикладних програм MATLAB. Цей програмний комплекс дозволяє проводити симуляції динамічних характеристик систем як за допомогою передавальних функцій, так і на основі векторно-матричних моделей, а також із застосуванням блоків моделювання в середовищі SIMULINK.

Для побудови моделі двомасової системи була створена блокова схема, що відтворює структуру алгоритмічної схеми, наведеної на рис. 3.4, та відповідає системі рівнянь стану (3.6). Розроблена схема моделі представлена на рис. 3.6.

Для спрощеного аналізу можна використовувати спрощену блокову модель системи, побудовану відповідно до спрощеної схеми двомасової системи на рис. 3.5 і рівнянь стану (3.7), яка наведена на рис. 3.7.

Окрім цього, для моделювання двомасової системи зручно застосовувати блок State-Space, що дозволяє безпосередньо задавати матриці стану та управління. Така реалізація показана на рис. 3.8, де в блоці State-Space вказуються імена матриць **A**, **B**, **C** і **D**, які формують модель двомасової системи.

Кількість вихідних координат, тобто розмірність вектора виходу $\vec{Y}(t)$, визначає кількість вікон Scope, у яких відображаються графіки перехідних процесів. Вікно налаштування матриць **A**, **B**, **C** і **D** для блоку State-Space наведене на рис. 3.9. При цьому, якщо необхідно відображати всі змінні стану, матриця виходу **C** приймається як одинична. У випадку, коли аналізуються лише окремі змінні стану, одиниці розташовуються по головній діагоналі лише в тих рядках, що відповідають потрібним змінним. Матриця **D** є нульовою.

У додатку Д наведено файл, що містить початкові параметри системи, включаючи матриці **A**, **B**, **C** і **D**.

На рис. 3.10–3.13 представлені графіки перехідних процесів основних змінних стану двомасової системи. Для зменшення обсягу інформації, що виводиться на екран, на моніторі відображено лише перші чотири змінні стану, які були роздруковані для подальшого аналізу.

Як можна спостерігати з графіків, динамічні характеристики двомасової системи демонструють слабо затухаючі коливання як по задаючій дії, так і по збурюючій. Це є типовим проявом пружності механічної частини системи, що враховує еластичність зв’язків між масами.

.

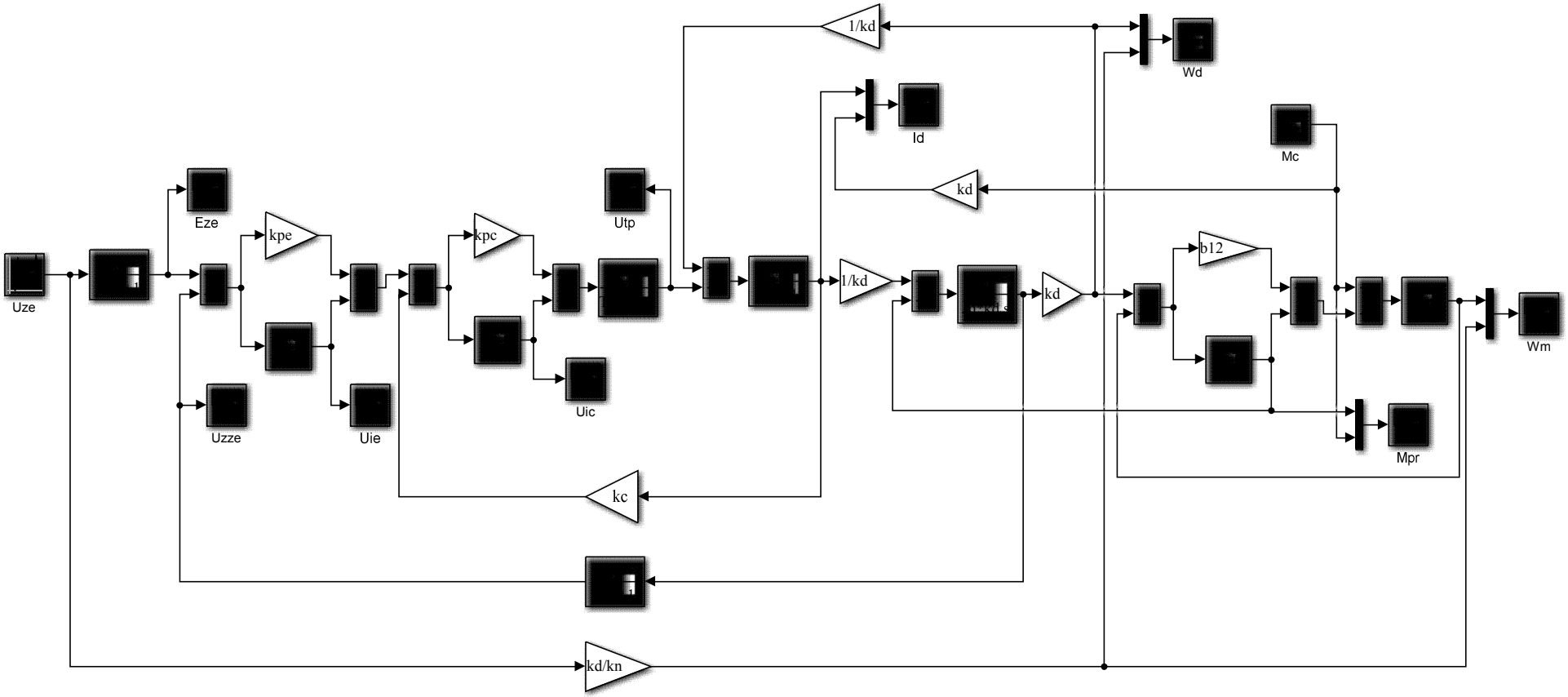


Рисунок 3.6 – Модель двомасової системи, розроблена в SIMULINK/ MATLAB

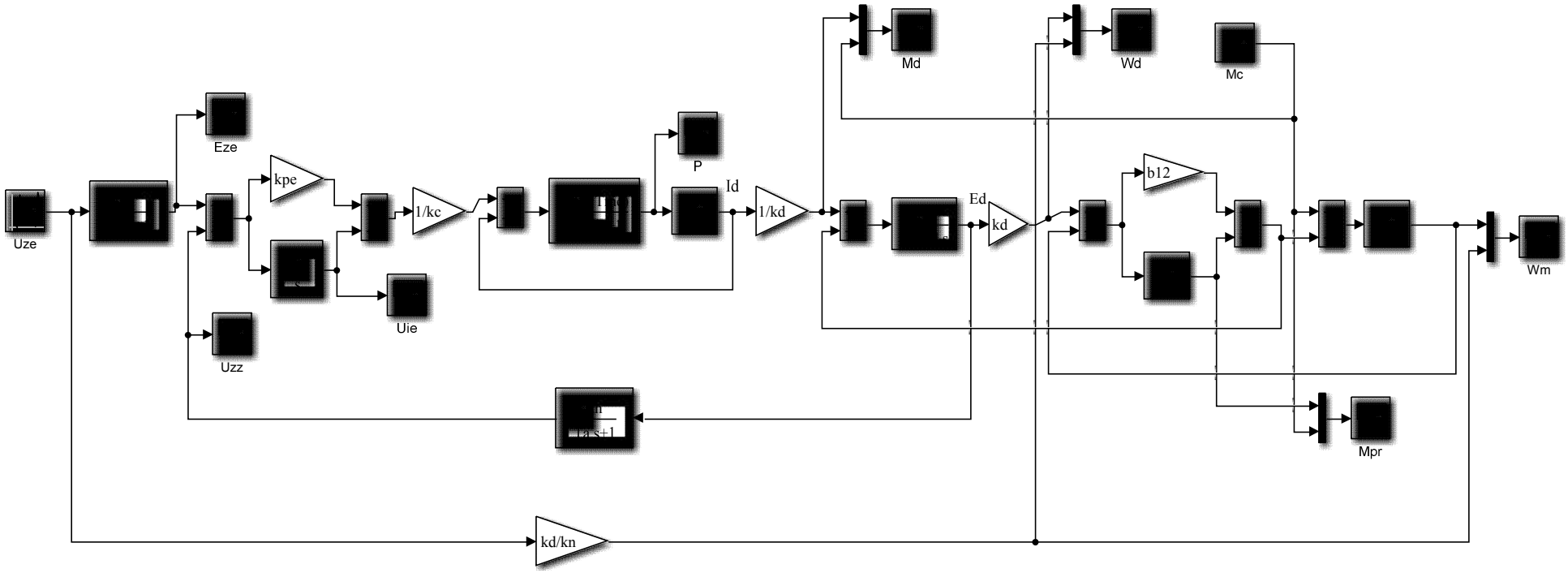


Рисунок 3.7 – Модель двомасової системи, розроблена в SIMULINK/MATLAB

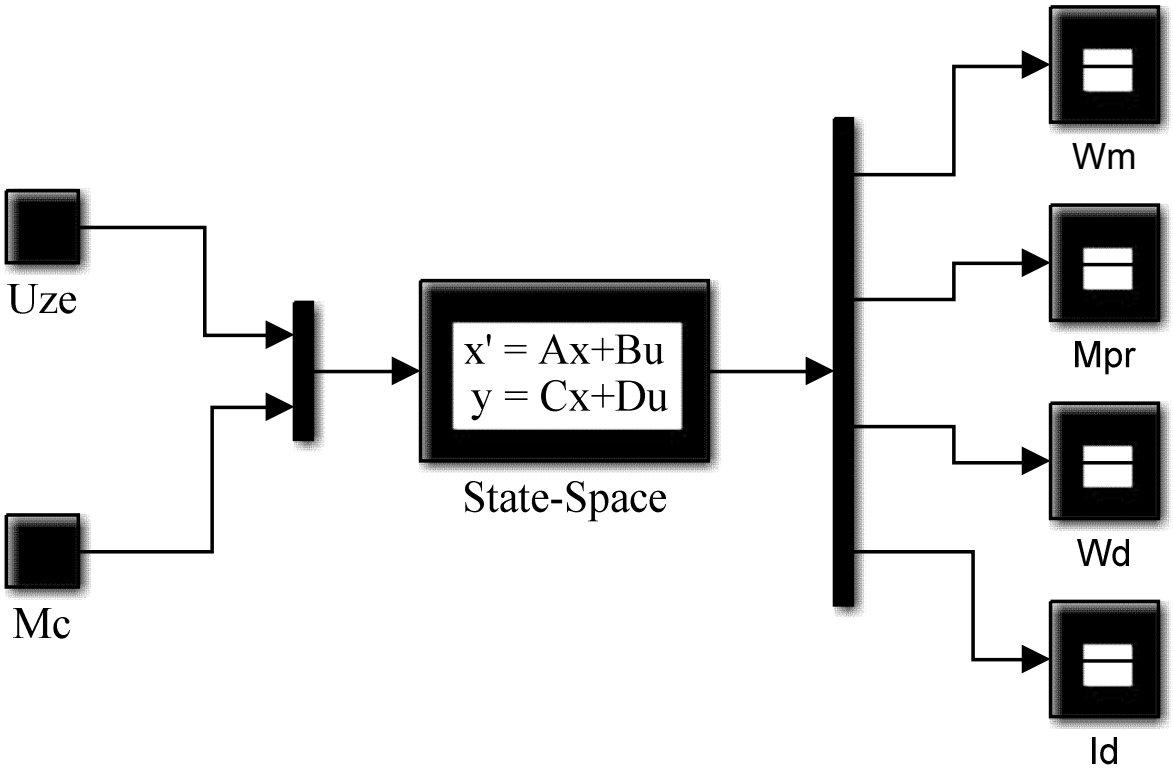


Рисунок 3.8 – Модель системи з використанням блоку State-Space

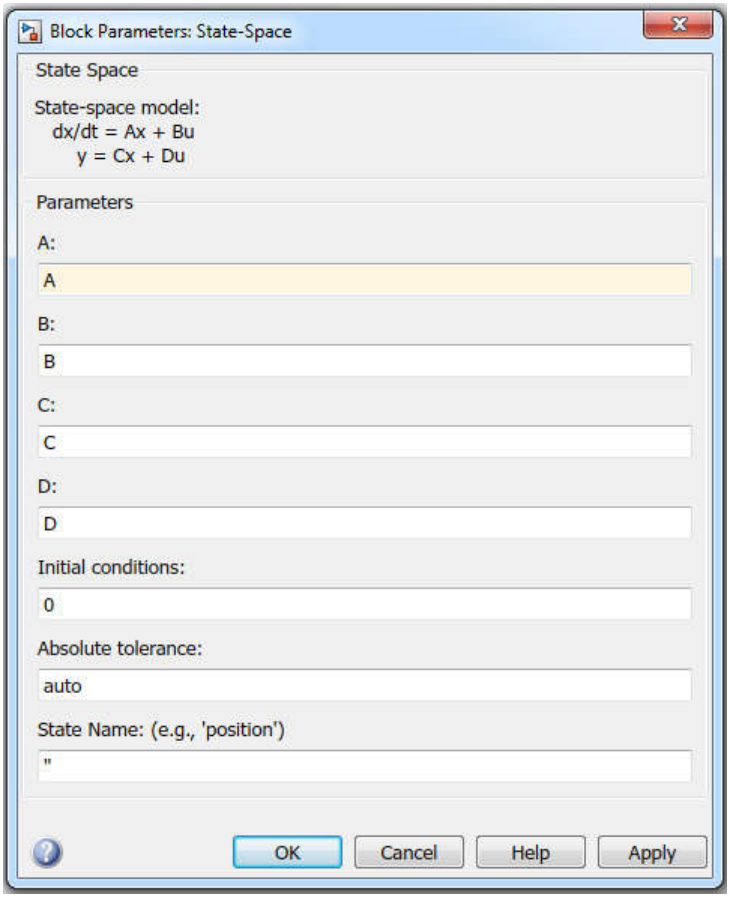
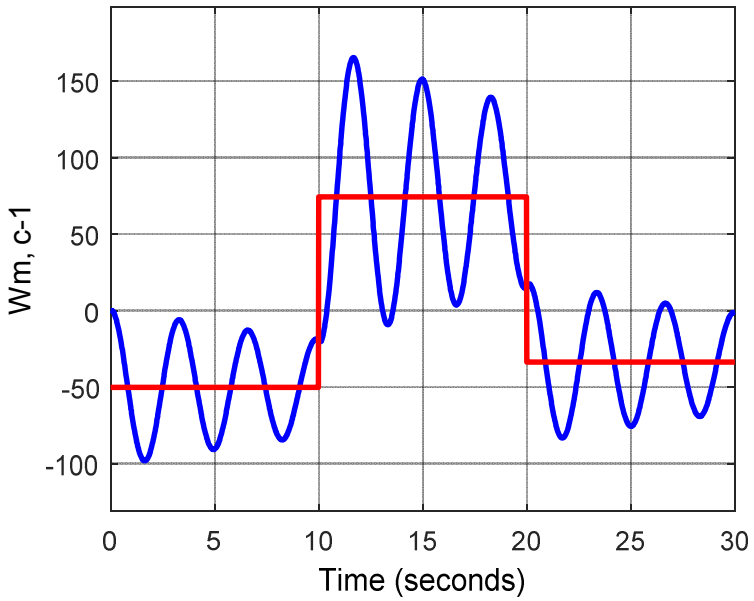
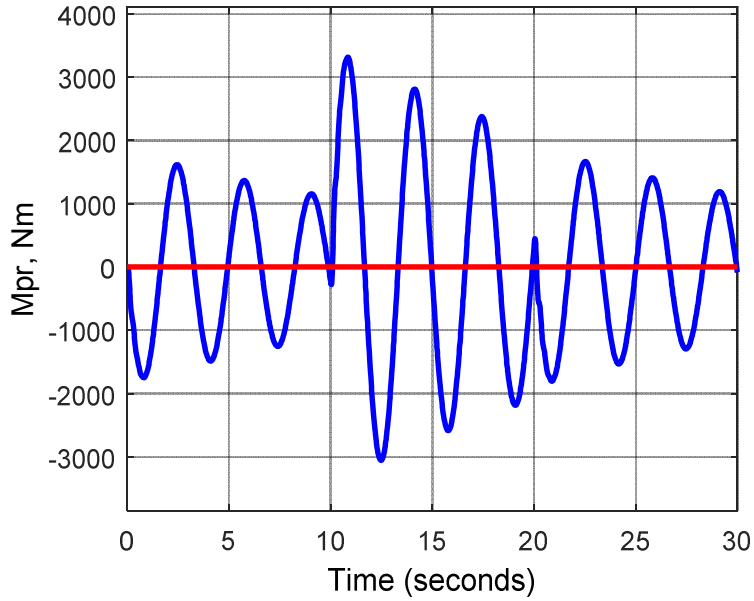


Рисунок 3.9 – Вікно налаштування матриць **A**, **B**, **C**, **D** блоку State-Spase

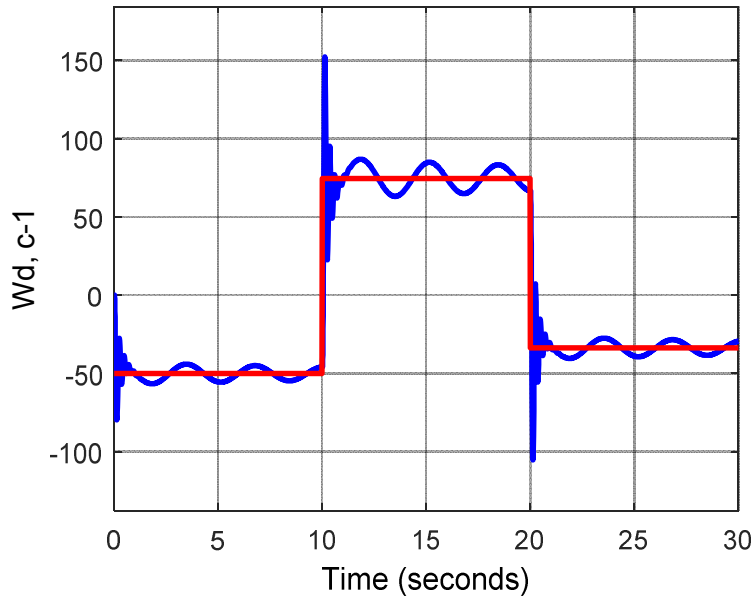


а)

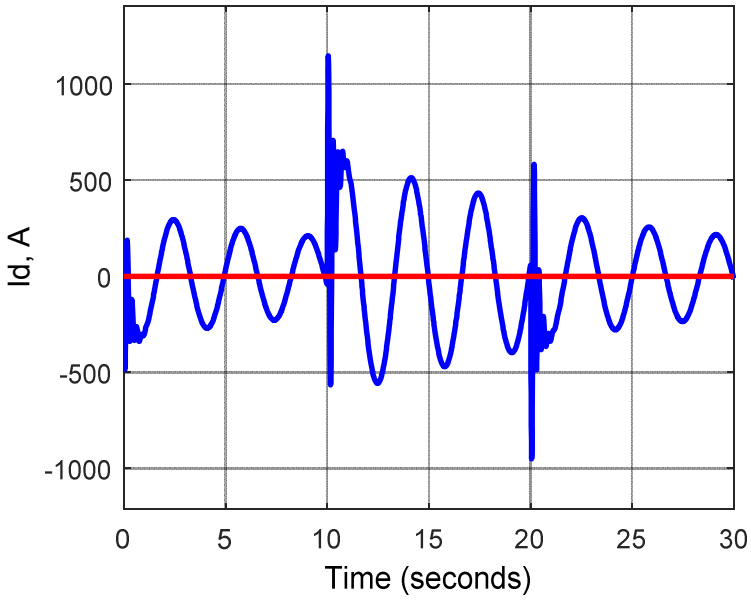


б)

Рисунок 3.10 – Перехідні процеси швидкості механізму швидкості механізму $\omega_{м} = f(t)$ (а) та пружного моменту $M_{пр} = f(t)$ (б) двомасової системи по задаючій дії

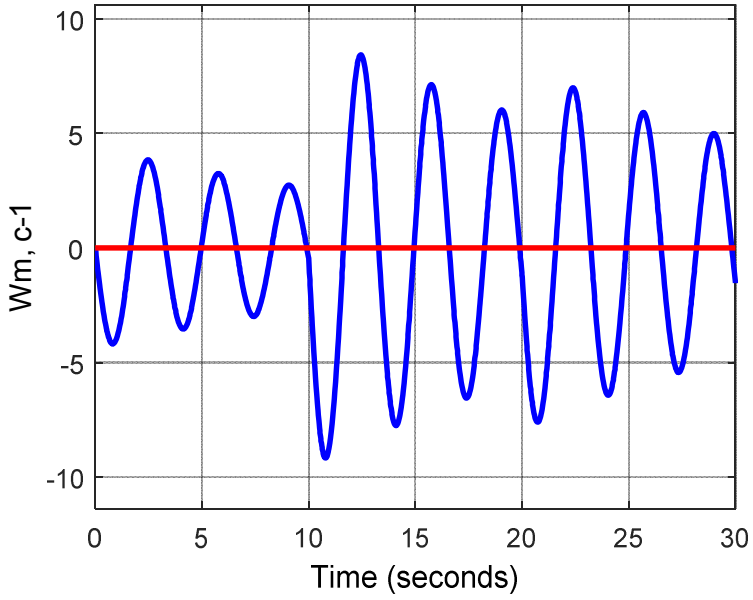


а)

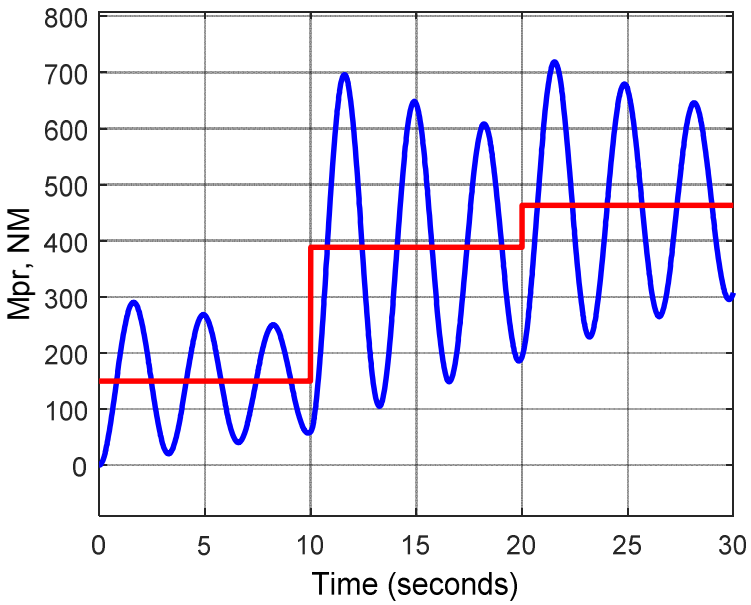


б)

Рисунок 3.11 – Перехідні процеси швидкості двигуна $\omega_д = f(t)$ (а) таструму двигуна $I_д = f(t)$ (б) двохмасової системи по задаючій дії

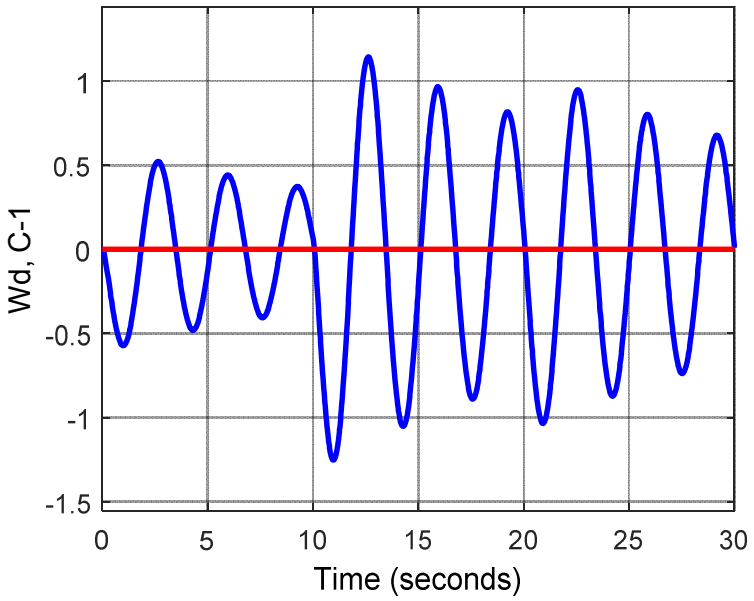


а)

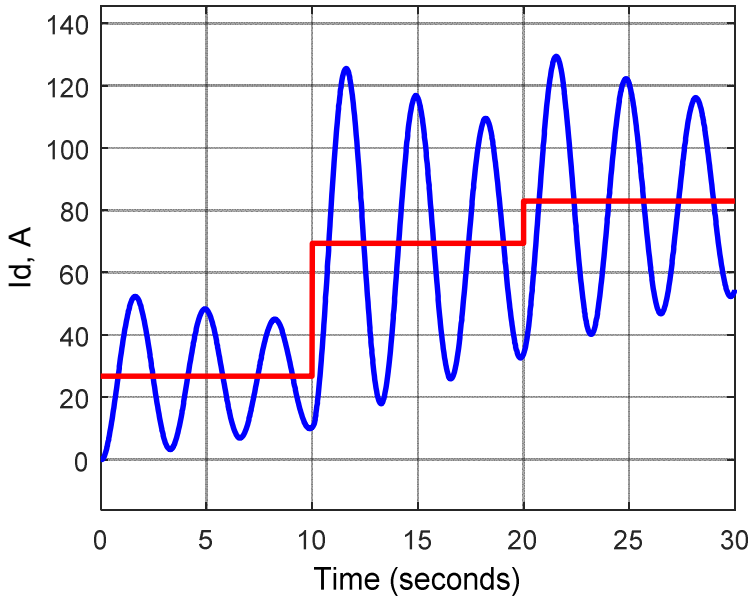


б)

Рисунок 3.12 Перехідні процеси швидкості механізму швидкості механізму $\omega_{м} = f(t)$ (а) та пружного моменту $M_{пр} = f(t)$ (б) двомасової системи по збурюючій дії



а)



б)

Рисунок 3.13 – Перехідні процеси швидкості двигуна $\omega_д = f(t)$ (а) таструму двигуна $I_д = f(t)$ (б) двохмасової системи по збурюючій дії

ВИСНОВКИ ДО РОЗДІЛУ 3

1. Проведено детальний аналіз динамічної поведінки механізму переміщення моста з урахуванням реальної кінцевої жорсткості механічного зв'язку між двигунами та мостовою конструкцією. Показано, що представлення механічної частини у вигляді жорсткої приведеної ланки відображає середній характер руху елементів і не враховує вплив пружних властивостей реальних з'єднань. Через обмежену жорсткість цих елементів механічна частина електроприводу є по суті пружною системою, де додаткова управляюча дія або збурююча дія (наприклад, момент від статичного навантаження) викликає коливання рухомих мас. Такі коливання призводять до підвищених максимальних навантажень на з'єднання та погіршують точність виконання необхідних переміщень і маніпуляцій. Встановлено, що кінематична схема механічної частини електроприводу з урахуванням пружності елементів може бути адекватно описана як двомасова система, що забезпечує більш точне моделювання динаміки руху.

2. Виконана розробка математичної моделі двомасової системи. Було отримано рівняння стану системи, включаючи їх представлення у векторно-матричній формі, визначено складові вектора стану (тобто змінні стану), сформульовано вектор зовнішніх впливів, а також визначено матриці стану **A** та управління **B.** Розроблено алгоритмічну схему двомасової системи та її спрощену версію, в якій контур струму оптимізований за модульним критерієм, що дозволяє точніше врахувати вплив пружних елементів на перехідні процеси системи.

3. Проведено комп'ютерне моделювання двомасової системи з використанням пакету прикладних програм MATLAB. Для цього в середовищі SIMULINK була побудована відповідна модель системи. Для тестування роботи системи на вхід подавався ступінчастий сигнал із випадковою амплітудою в межах від +24 В до 24 В, що відповідає зміні швидкості у межах номінальних параметрів. Аналіз результатів моделювання показав, що перехідні процеси

змінних стану двомасової системи мають виражений коливальний характер, що вказує на необхідність подальшої оптимізації контурів регулювання або введення компенсуючих пристроїв для згладжування коливань і підвищення якості динаміки системи.

РОЗДІЛ 4
СИНТЕЗ НЕЙРОМЕРЕЖЕВОЇ СИСТЕМИ УПРАВЛІННЯ

4.1 Регулятори, реалізовані засобами пакета Neural Network Toolbox у середовищі MATLAB

Процес побудови нейромережевої системи управління виконувався за допомогою інструментів пакета Neural Network Toolbox, що входить до складу MATLAB. Далі наведено короткий огляд можливостей цього пакета, описано послідовність створення нейронного регулятора та подано результати моделювання роботи нейромережевої системи керування.

У MATLAB розглянуто три основні архітектури нейронних мереж, реалізованих у зазначеному пакеті прикладних програм, які використовуються при побудові таких типів регуляторів:

- регулятор із прогнозуванням майбутніх станів системи (NN Predictive Controller);
- регулятор, що ґрунтується на моделі авторегресії з ковзним середнім (NARMA-L2 Controller);
- регулятор на основі еталонної моделі (Model Reference Controller).

Використання нейронних мереж у задачах автоматичного керування передбачає поділ процесу проектування на два основних етапи:

- етап ідентифікації моделі об’єкта;
- етап синтезу закону керування.

Під час ідентифікації будується нейромережева модель об’єкта керування, яка надалі використовується для розробки потрібного регулятора. Для всіх трьох згаданих архітектур застосовується однакова процедура побудови моделі, однак принципи синтезу регуляторів між собою істотно відрізняються.

У випадку регулятора, що працює з прогнозом, нейронна модель об’єкта призначена для оцінювання його майбутніх станів, а оптимізаційний алгоритм підбирає таке керуюче діяння, яке мінімізує відхилення між бажаною та реальною траєкторіями вихідної величини.

Для регулятора типу NARMA-L2 структура керування являє собою модифіковану версію моделі самого об’єкта і будується на основі авторегресії з ковзним середнім.

У випадку використання еталонної моделі регулятор являє собою нейронну мережу, що навчається таким чином, щоб змусити реальний об’єкт повторювати поведінку вибраної еталонної системи. При цьому модель об’єкта задіюється безпосередньо в процесі налаштування параметрів регулятора.

Динамічні моделі систем керування з нейромережевими регуляторами зібрані в окремому підрозділі *Control Systems* бібліотеки Neural Network Toolbox (див. рис. 4.1).

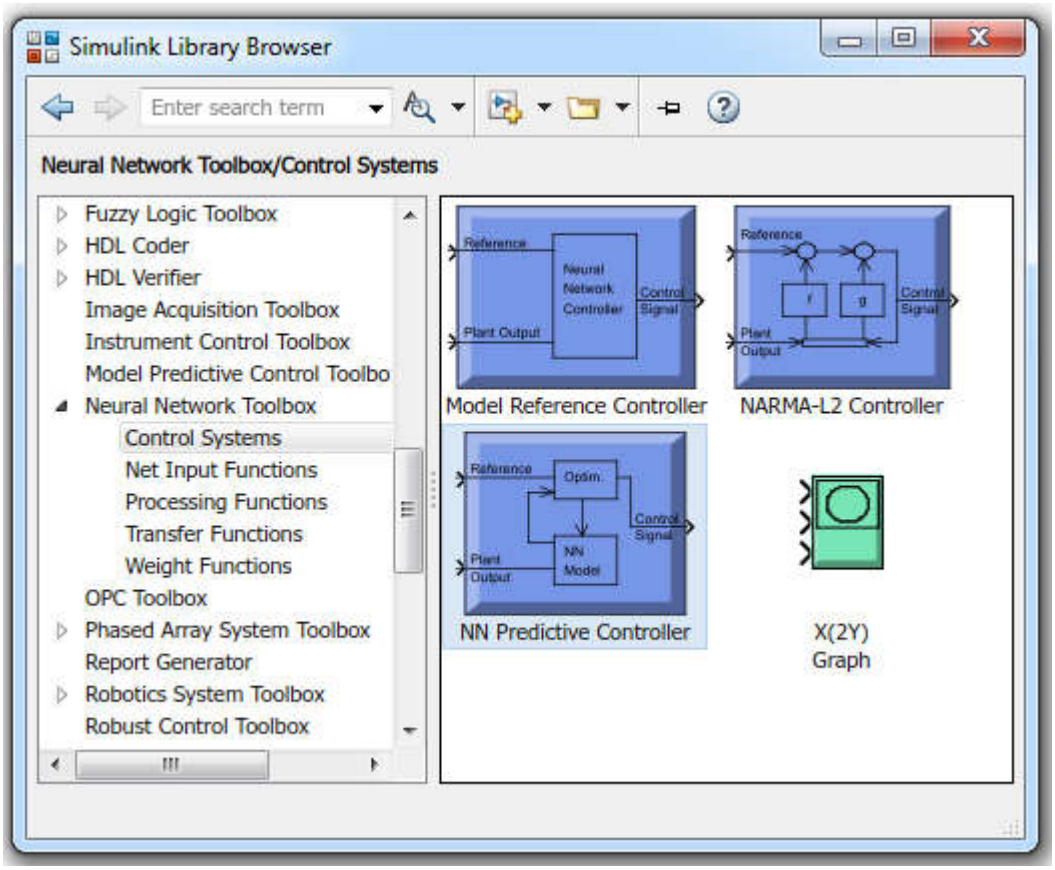


Рисунок 4.1 – Набір блоків Neural Network Blocksets підсистеми Control Systems середовища SIMULINK

Регулятор із прогнозуванням. У цьому типі регулятора нейромережева модель об’єкта використовується для прогнозу того, як система поводитиметься при різних можливих керуючих впливах. На основі цих прогнозів оптиміза-

ційний алгоритм вибирає такі сигнали керування, які забезпечують мінімальну похибку між заданою та фактичною траєкторією вихідного сигналу моделі. Формування нейромережевої моделі здійснюється окремо: мережу навчають пакетно, застосовуючи один із доступних навчальних алгоритмів. Недоліком цього підходу є значні обчислювальні витрати, оскільки оптимізація проводиться на кожному кроці формування керуючої дії.

Регулятор NARMA-L2 (на основі авторегресійної моделі з ковзним середнім). Серед трьох архітектур цей варіант є найбільш економним з точки зору обчислювальних ресурсів. По суті, регулятор є модифікацією отриманої на попередньому етапі нейромережевої моделі об'єкта. Основним недоліком методу є необхідність подавати модель у формі простору станів у канонічному вигляді зі строго визначеною структурою супровідної матриці. Це може спричинювати додаткові неточності в обчисленнях.

Регулятор з еталонною моделлю. Обсяг обчислень, необхідний для такого регулятора, приблизно відповідає обсягу, потрібному для NARMA-L2. Однак структура цього регулятора складніша: потрібно одночасно навчити нейромережу, що відтворює поведінку об'єкта, і нейромережу, яка виконує функції регулятора. Процес навчання складніший, оскільки використовує динамічний варіант алгоритму зворотного поширення помилки. Головною перевагою підходу є його універсальність – регулятор на базі еталонної моделі можна адаптувати до багатьох типів об'єктів керування.

Для електроприводу, що аналізується у цій роботі, було обрано саме регулятор із прогнозуванням. Варіанти на основі NARMA-L2 та еталонної моделі не забезпечили достатнього рівня якості керування в процесі моделювання.

4.2 Принцип формування прогнозуючого регулятора NN Predictive Controller

Регулятор типу NN Predictive Controller, реалізований у складі пакету Neural Network Toolbox системи MATLAB, ґрунтується на використанні ней-

ромережевої моделі нелінійного об’єкта керування. Така модель забезпечує можливість передбачити подальший розвиток процесів у системі, а також дозволяє сформувати оптимальний керуючий вплив на визначеному інтервалі прогнозування.

4.2.1 Процедура ідентифікації об’єкта керування

Схему модуля ідентифікації показано на рис. 4.2. До складу цього модуля входить нейромережева модель об’єкта, яка навчається у відокремленому (автономному) режимі. Метою навчання є мінімізація розбіжності між реакцією реального об’єкта та відповіддю моделі на подану послідовність тестових впливів.

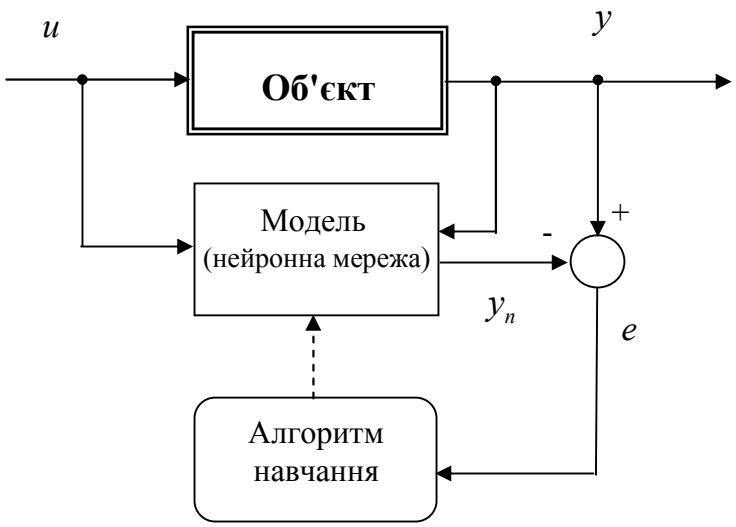


Рисунок 4.2 – Структурна схема підсистеми ідентифікації

Архітектуру нейромережі, що використовується для моделювання об’єкта, наведено на рис. 4.3. Вона є дворівневою (двошаровою) і включає лінії затримки, які дозволяють зберігати попередні значення сигналів на вході та виході об’єкта. Це забезпечує можливість прогнозувати майбутні значення вихідної координати.

Параметри мережі налаштовуються автономно, на основі набору експериментальних даних, отриманих у процесі тестування системи. Для навчання може застосовуватися будь-який зі стандартних алгоритмів нейромережевого

навчання. У даному дослідженні використано метод Левенберга–Марквардта, який в Neural Network Toolbox реалізований за допомогою М-функції **trainlm**.

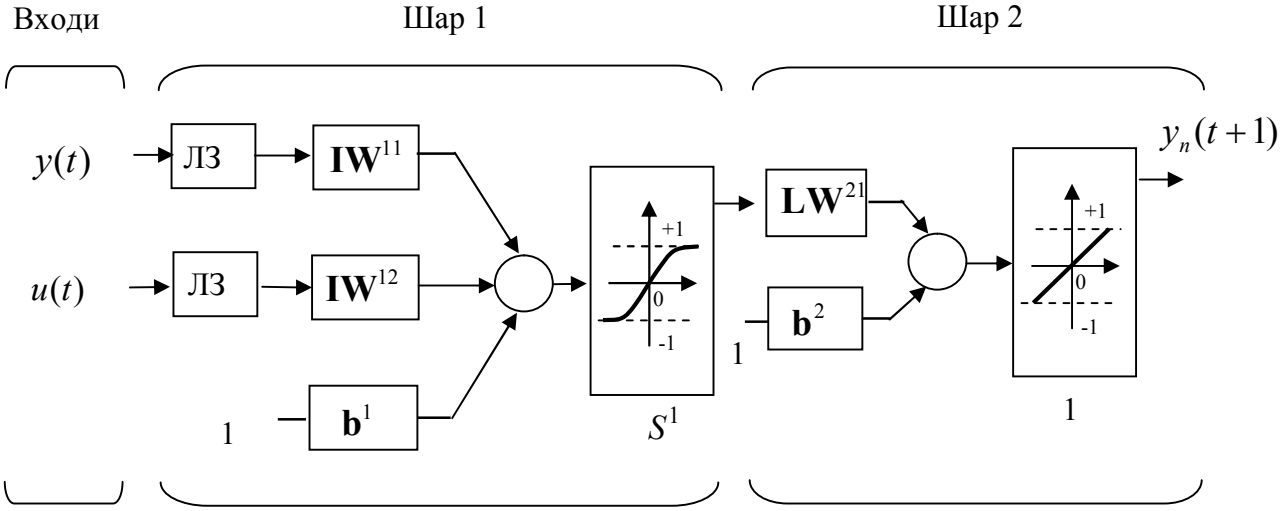


Рисунок 4.3 – Структура нейронної мережі, що моделює об’єкт керування

4.2.2 Принцип функціонування прогнозуючої системи керування

Алгоритм управління на основі прогнозу працює за ідеологією «ковзного горизонту». Це означає, що нейромережева модель об’єкта безперервно обчислює ймовірну реакцію системи на деякому майбутньому відрізку часу. Отриманий прогноз передається в оптимізаційний модуль, який обирає таку керуючу дію, що мінімізує критерій якості:

$$J = \sum_{j=N_1}^{N_2} \left[y_r(t+j) - y_m(t+j) \right]^2 + \rho \sum_{j=1}^{N_4} \left[u'(t+j-1) - u'(t+j-2) \right]^2, \qquad (4.1)$$

Тут коефіцієнти N_1, N_2 і N_4 визначають межі інтервалу, на якому аналізуються відхилення вихідного сигналу та енергетичні витрати на формування управління. Змінна u' – пробний керуючий вплив, y_r – бажана траєкторія, а y_m – прогнозована реакція моделі. Параметр ρ регулює вагу енергетичної складової у функції якості.

Узагальнену блок-схему системи із застосуванням прогнозуючого принципу наведено на рис. 4.4. Вона складається з нейронної моделі об’єкта та оп-

тимізаційного модуля. Останній знаходить значення u', які мінімізують критерій якості, а сформований сигнал керування надходить безпосередньо на об'єкт.

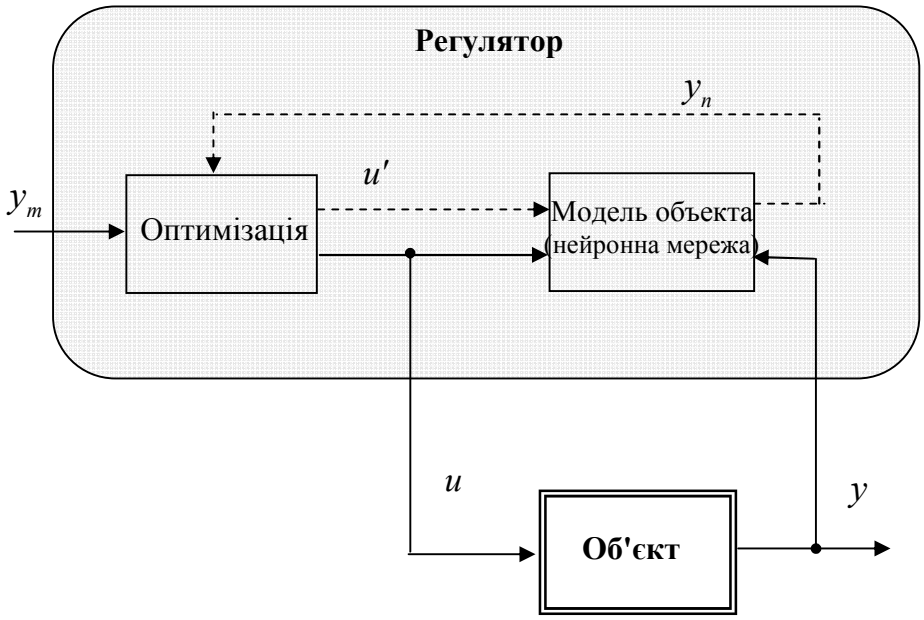


Рисунок 4.4 – Узагальнена схема керування з використанням прогнозуючого регулятора

4.3 Проєктування прогнозуючого регулятора NN Predictive Controller у середовищі MATLAB

Під час побудови нейромережевого регулятора типу NN Predictive Controller використовується набір спеціалізованих функцій і моделей, що входять до складу каталогу *toolbox/nnet/nncontrol* пакету Simulink. До цього набору належать такі групи програмних засобів:

1. Функції одновимірної оптимізації

Ці алгоритми застосовуються під час пошуку оптимального керуючого впливу в процесі роботи регулятора:

- **Csrchbac** – алгоритм пошуку за схемою «зворотного проходу» (backtracking).
- **Csrchbre** – модифікований метод Брента, який поєднує техніку золотого перетину та квадратичну інтерполяцію.

- **Csrchcha** – метод кубічної інтерполяції Чараламбуса.
- **Csrchgol** – класичний метод золотого перетину.
- **Csrchhyb** – комбінований алгоритм, що використовує бісекцію та кубічну інтерполяцію.

2. Функції, що реалізують механізм керування з прогнозуванням

Ці М-функції забезпечують обчислення критеріїв якості та роботу оптимізаційних алгоритмів:

- **Calcjjdjj** – оцінювання критерію якості та його градієнта;
- **Predopt** – основний модуль оптимізації прогнозуючого регулятора;
- **Dyduvar** – розрахунок часткових похідних вихідних сигналів за входами.

3. Моделі та графічні додатки Simulink

- **Predcstr** – графічний інтерфейс користувача (GUI) для роботи з регулятором типу Predictive Controller;
- **Prest3sim2** – нейромережева модель об’єкта керування, що застосовується функцією *deport* для формування прогнозу поведінки системи.

4. Допоміжні програмні модулі

- **Nncontrolutil** – набір службових функцій, що забезпечують взаємодію регулятора з внутрішніми компонентами Simulink;
- **Nnident.m** – ключова функція, яка використовується під час етапу ідентифікації об’єкта. Розташована у підкаталозі *private*, забезпечує створення навчальної вибірки, формування нейронної мережі та процес її навчання через зручний GUI.

Після підготовки необхідних даних у середовищі Simulink формується модель системи керування, структура якої наведена на рисунку 4.5.

У представленій конфігурації системи міститься підсистема, що моделює об'єкт керування (блок *Subsystem*), модуль нейромережевого регулятора (*NN Predictive Controller*), генератор випадкового еталонного ступінчастого сигналу (*Random Reference*), а також елементи для візуалізації результатів.

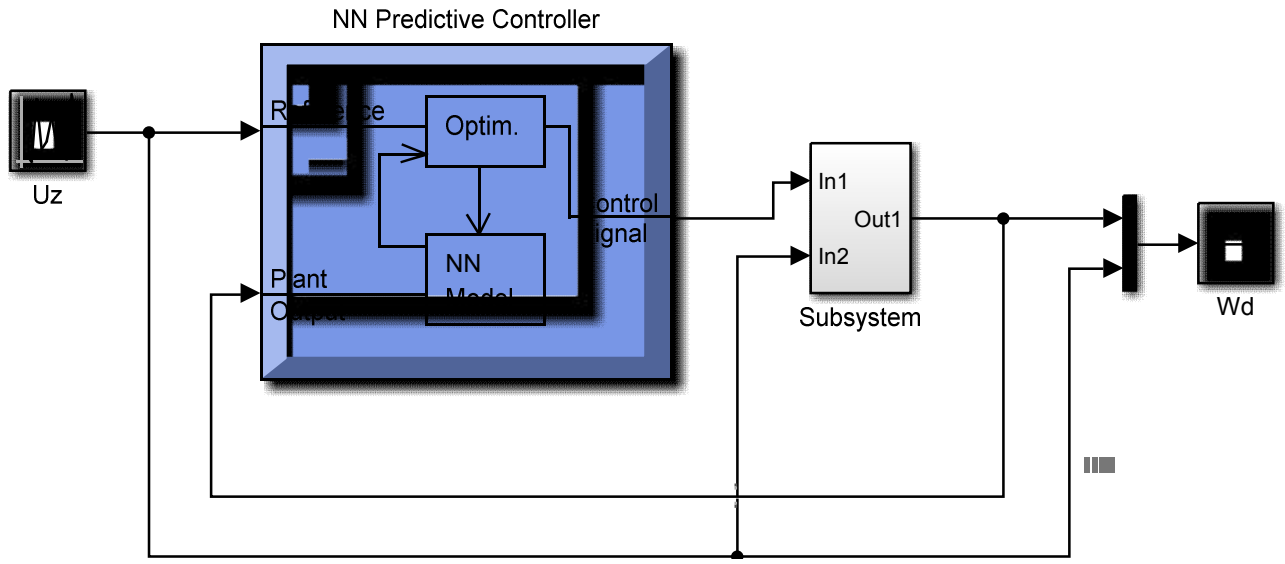


Рисунок 4.5 – Структурна модель системи керування з нейрорегулятором NN Predictive Controller

Важливо зазначити, що дана схема одночасно виконує роль стандартної моделі середовища SIMULINK і працює як графічний інтерфейс користувача (GUI), забезпечуючи безпосередню взаємодію з налаштуваннями системи.

Структура внутрішньої організації нейромережевого регулятора NN Predictive Controller подана на рис. 4.6. Вона відкривається у спеціальному вікні після вибору команди **Look Under Mask** для відповідного блоку.

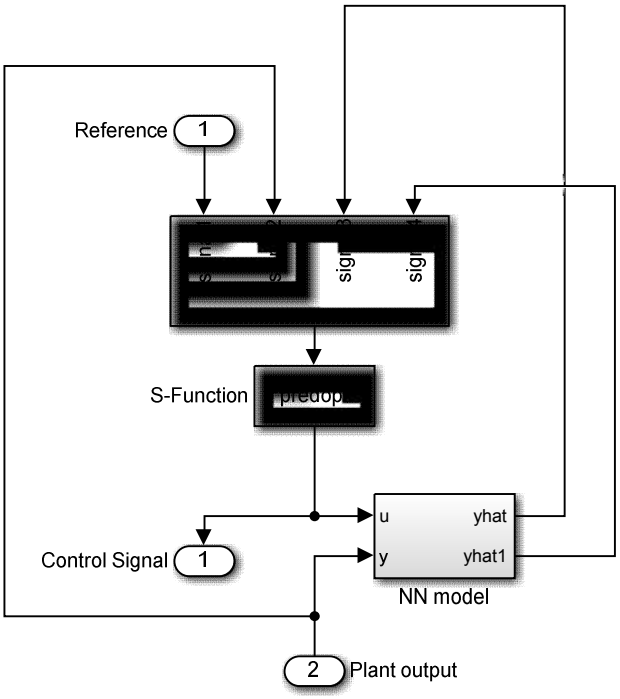


Рисунок 4.6 – Структурна схема нейрорегулятора

Розробка регулятора розпочинається з активації блоку NN Predictive Controller. Після цього відкривається інтерфейс, зображений на рис. 4.7, який слугує панеллю налаштування параметрів. У верхній частині вікна відображається інформаційне повідомлення *«Perform plant*

identification before controller configuration», яке підказує, що перед подальшими налаштуваннями обов'язково потрібно провести етап ідентифікації – сформувати нейромережеву модель керованого об'єкта за допомогою процедури **Plant Identification**.

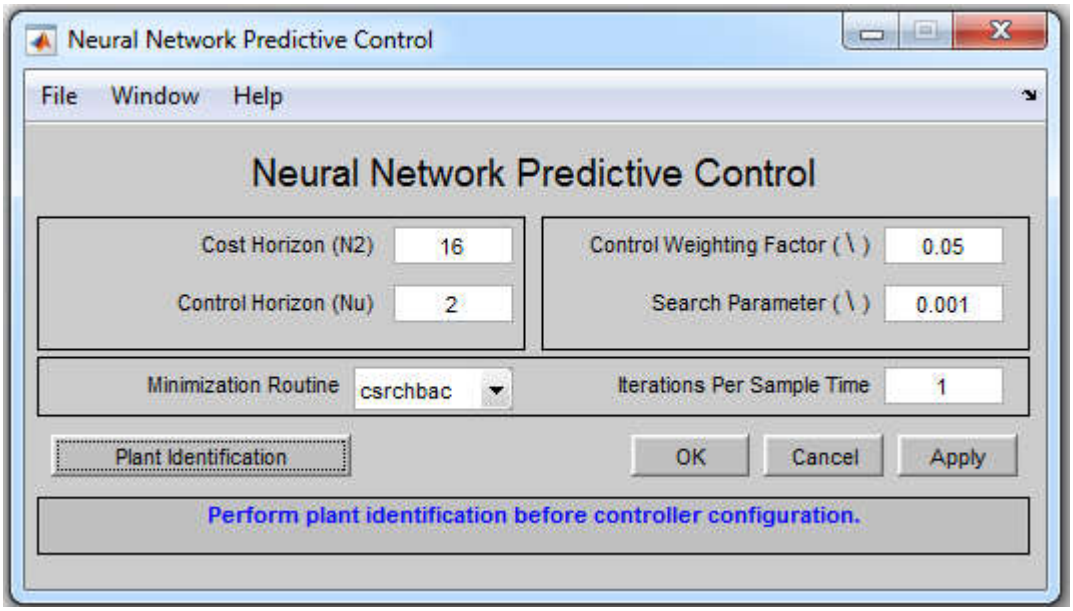


Рисунок 4.7 – Інтерфейс налаштування параметрів регулятора

На рис. 4.8 показано вікно модуля Plant Identification. Воно є універсальним інструментом, що дозволяє створювати нейромережеві моделі для будь-яких динамічних систем, описаних у SIMULINK. Для даної роботи таким об'єктом виступає модель двомасового механізму.

Під час ідентифікації формується нейронна мережа, яка повинна відтворювати поведінку реального об'єкта, оскільки саме ця модель буде використана в подальшому для налаштування та оптимізації регулятора. Тому етап створення моделі має виконуватися до розрахунків самого регулятора.

Процедура ідентифікації вимагає попереднього задання низки параметрів, які налаштовуються через вікно **Plant Identification**. Це вікно поділене на кілька логічних секцій, кожна з яких відповідає за певний аспект формування нейромережевої моделі.

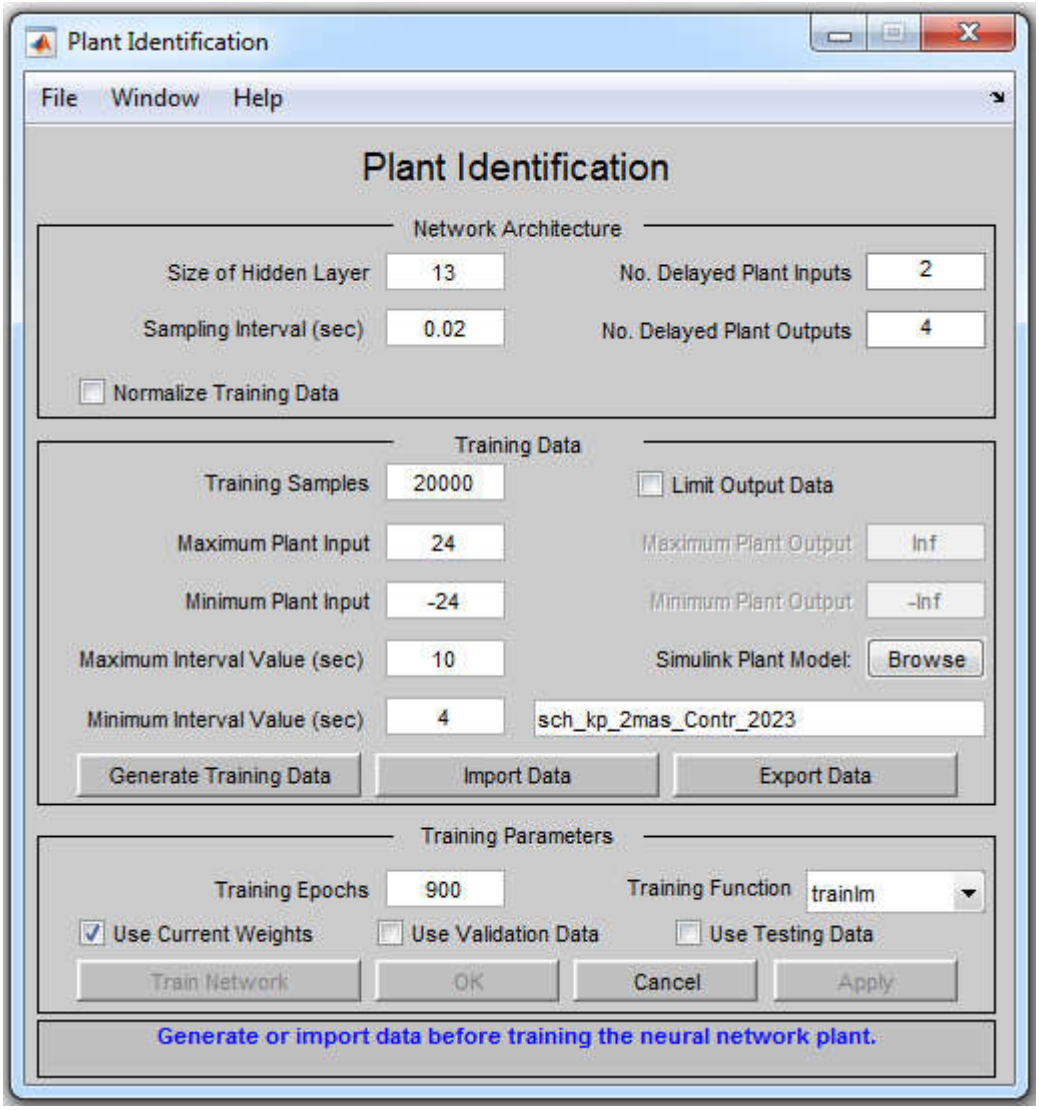


Рисунок 4.8 – Вікно процедури ідентифікації об’єкта керування

Опис вікна Plant Identification

Секція Network Architecture (Архітектура мережі).

У цій частині задаються параметри, які визначають структуру нейронної мережі, що використовується для ідентифікації та подальшого відтворення динаміки досліджуваної системи. Коректний вибір налаштувань у секції **Network Architecture** є принципово важливим, оскільки саме від них залежить, наскільки точно нейромережа здатна описати поведінку об’єкта.

Параметр Size of Hidden Layer (кількість нейронів прихованого шару). Цей параметр вказує, скільки нейронів міститиметься у прихованому шарі мережі, яка застосовується для моделювання або ідентифікації об’єкта. Вибір

розміру прихованого шару є одним із ключових рішень при проектуванні нейромережі:

- збільшення кількості нейронів, як правило, дозволяє мережі краще відтворювати складні, нелінійні залежності в даних;
- однак занадто великий прихований шар може спричинити перенавчання (overfitting), коли мережа добре відтворює навчальні дані, але погано узагальнює поведінку на нових вхідних впливах.

Тому розмір прихованого шару зазвичай розглядають як гіперпараметр, який підбирають експериментально, орієнтуючись на якість моделювання для конкретної задачі.

Параметр Sampling Interval (sec) (інтервал дискретизації, с). Цей параметр задає часовий інтервал між двома послідовними вимірюваннями сигналів, що використовуються для ідентифікації. Іншими словами, це період, з яким знімаються дані з моделі або реального об’єкта.

Вибір інтервалу дискретизації має суттєвий вплив на:

- деталізацію та роздільну здатність отриманих даних;
- точність подальшого відтворення динаміки системи.

Якщо інтервал вибірки занадто великий, модель може «не побачити» швидких змін у процесі, а надто малий інтервал призводить до надлишку даних і зростання обчислювальних витрат. Тому при заданні Sampling Interval необхідно враховувати швидкодію об’єкта і вимоги до точності результатів ідентифікації.

No. Delayed Plant Inputs (кількість затриманих входів). Цей параметр визначає, скільки попередніх значень вхідних сигналів системи беруться до уваги під час ідентифікації. Урахування запізнених входів дає змогу моделі відобразити наявні у системі затримки та більш коректно відтворити її динаміку.

Такі затримки часто зустрічаються в реальних керованих процесах, де реакція на вхід проявляється із певним часовим відставанням. Тому коректне встановлення параметра **No. Delayed Plant Inputs** є важливим для отримання

точної моделі, особливо коли відомо, що між дією на об’єкт і його реакцією існує часовий лаг.

Правильне задання цього параметра дає можливість адаптувати модель до реальних умов роботи системи та підвищити якість процесу ідентифікації.

No. Delayed Plant Outputs (кількість затриманих виходів). Параметр указує кількість попередніх значень вихідних сигналів, які залучаються до формування моделі. Використання запізнених виходів дозволяє врахувати затримки у формуванні відгуку системи, що суттєво впливає на точність моделювання її поведінки.

Цей параметр визначає, наскільки глибоко модель аналізує історію вихідних значень. Якщо система має значні внутрішні затримки, то включення більшої кількості запізнених виходів може суттєво підвищити адекватність отриманої моделі.

Оптимальний вибір No. Delayed Plant Outputs залежить від властивостей конкретного об’єкта та рівня точності, якого потрібно досягти під час ідентифікації.

Normalize Training Data (нормалізація навчальних даних). Цей пункт відповідає за попереднє масштабування вхідних і вихідних сигналів перед їх використанням у процесі навчання нейромережевої моделі. Вмикання цього параметра означає, що всі дані будуть приведені до певного стандартного діапазону чи форми – наприклад, до інтервалу [0;1], або ж віднормовані за середнім значенням і стандартним відхиленням.

Якщо нормалізація активована, система автоматично перетворює дані та подає на вхід мережі більш узгоджені значення. Якщо ж цей пункт вимкнено, навчання виконується на сирих даних без попередньої обробки.

Основне призначення нормалізації – забезпечити стабільніше та швидше навчання нейронної мережі, особливо у випадках, коли окремі змінні мають різні масштаби або різко відрізняються за величинами. Коректно виконане масштабування також позитивно впливає на точність і якість побудованої моделі.

Доцільність увімкнення **Normalize Training Data** визначається особливостями конкретної системи та вимогами до процесу її ідентифікації.

Секція Training Data (навчальні дані). У розділі *Training Data* задається набір сигналів, на основі яких формується нейромережева модель досліджуваної системи. До таких даних входять часові послідовності вхідних впливів та відповідних виходів об’єкта, що дозволяє мережі відтворити закономірності та динаміку реального процесу.

Головне призначення цього набору – забезпечити мережу достатньою кількістю інформації для коректного навчання, щоб побудована модель могла описувати поведінку системи в різних умовах. Як правило, до Training Data включають результати роботи об’єкта в різних режимах, що підвищує здатність моделі враховувати широке коло можливих станів та реакцій.

Під час ідентифікації саме на цій вибірці нейронна мережа шукає відповідність між вхідними і вихідними сигналами, що визначає точність отриманої моделі. Тому якість і повнота навчальних даних безпосередньо впливають на надійність побудованої моделі.

Training samples (кількість навчальних точок). Цей параметр визначає розмір навчальної вибірки – тобто число вимірів, використаних для побудови моделі. Від його значення залежить, наскільки глибоко та точно нейронна мережа зможе відтворити характеристики керованого об’єкта: надто мала вибірка може призвести до неточностей, тоді як достатній обсяг даних забезпечує більш адекватну ідентифікацію.

Maximum Plant Input (верхня межа вхідного сигналу). Цей параметр визначає найбільше допустиме значення сигналу, що подається на вхід системи під час процедури ідентифікації. Фактично він задає верхню границю амплітуди вхідного впливу, який використовуватиметься у навчальній вибірці.

Коректне встановлення цього обмеження дає змогу забезпечити, щоб модель навчалася лише в межах реальних або фізично обґрунтованих режимів роботи об’єкта. Якщо допустимий максимум буде вибрано занадто великим, то

мережа може навчитися на значеннях, які не відповідають реальним умовам експлуатації, що негативно позначиться на точності прогнозування.

Minimum Plant Input (нижня межа вхідного сигналу). Параметр визначає мінімально можливе значення керуючого сигналу, який може подаватися під час збирання даних для ідентифікації. Це нижня межа діапазону вхідних впливів, що враховується у процесі навчання нейромережевої моделі.

Разом із максимальним значенням цей параметр формує допустимий інтервал вхідних сигналів, у межах якого система вважається працездатною. Невдало вибрані межі можуть спричинити зниження якості моделі, оскільки мережа або не отримає достатньо різноманітних даних, або навпаки – опрацьовуватиме нереалістичні значення сигналів.

Правильно заданий діапазон входів забезпечує адекватність моделі та її здатність точно передбачати реакцію системи в реальних умовах.

Maximum Interval Value (sec) – верхня межа інтервалу часу. Цей параметр задає максимальну тривалість інтервалу у секундах, протягом якого використовуються вхідні та вихідні сигнали для ідентифікації динаміки системи. По суті, він визначає максимально допустиму довжину часу, на яку поширюється навчальний набір даних під час побудови моделі.

Встановлення цього значення дозволяє обмежити часовий діапазон аналізованої інформації. Це особливо корисно, коли деякі періоди роботи системи менш значущі для процесу ідентифікації, а дані в інших проміжках є ключовими. Вибір параметра Maximum Interval Value зазвичай враховує специфіку досліджуваного об’єкта та тривалість циклів його роботи, обмежуючи аналіз лише необхідним часовим відрізком.

Minimum Interval Value (sec) (нижня межа інтервалу часу). Цей параметр визначає мінімальну тривалість інтервалу у секундах, протягом якого враховуються вхідні та вихідні дані для ідентифікації системи. Іншими словами, він задає коротший допустимий часовий проміжок для навчальної вибірки.

Налаштування Minimum Interval Value дозволяє контролювати, які частини часового ряду вважати значущими для побудови моделі. Використання цьо-

го параметра допомагає виключати надто короткі або неінформативні інтервали, забезпечуючи, щоб у навчанні брали участь тільки дані з необхідною тривалістю або значущістю. Правильне встановлення мінімального інтервалу дозволяє оптимізувати ідентифікаційний процес відповідно до особливостей та вимог конкретної системи.

Limit Output Data (обмеження вихідних даних). Цей параметр активує вікно управління, яке дозволяє регулювати обсяг вихідних даних, що використовуються під час ідентифікації. При ввімкненні цього вікна стають доступними два додаткові поля для редагування:

- Maximum Plant Output – максимальне значення вихідного сигналу.
- Minimum Plant Output – мінімальне значення вихідного сигналу.

Параметри Maximum Plant Output та Minimum Plant Output визначають обмеження вихідних даних, які беруть участь у процесі навчання моделі та аналізу динаміки системи. Це особливо корисно, коли потрібно сконцентруватися на певному діапазоні вихідних величин або дослідити специфічну поведінку системи протягом конкретного проміжку часу.

Використання цих обмежень дозволяє зменшити обсяг оброблюваних даних, спростити процес навчання та зробити ідентифікацію моделі більш ефективною.

Simulink Plant Model (модель об’єкта в Simulink). Цей параметр задає модель керованого об’єкта в середовищі Simulink, визначаючи її вхідні та вихідні порти, які використовуються для побудови нейромережевої моделі. Вибирається конкретна модель об’єкта або системи для ідентифікації. За допомогою кнопки Browser здійснюється вибір моделі Simulink; у цій роботі використовується схема моделі, наведена на рис. 4.9.

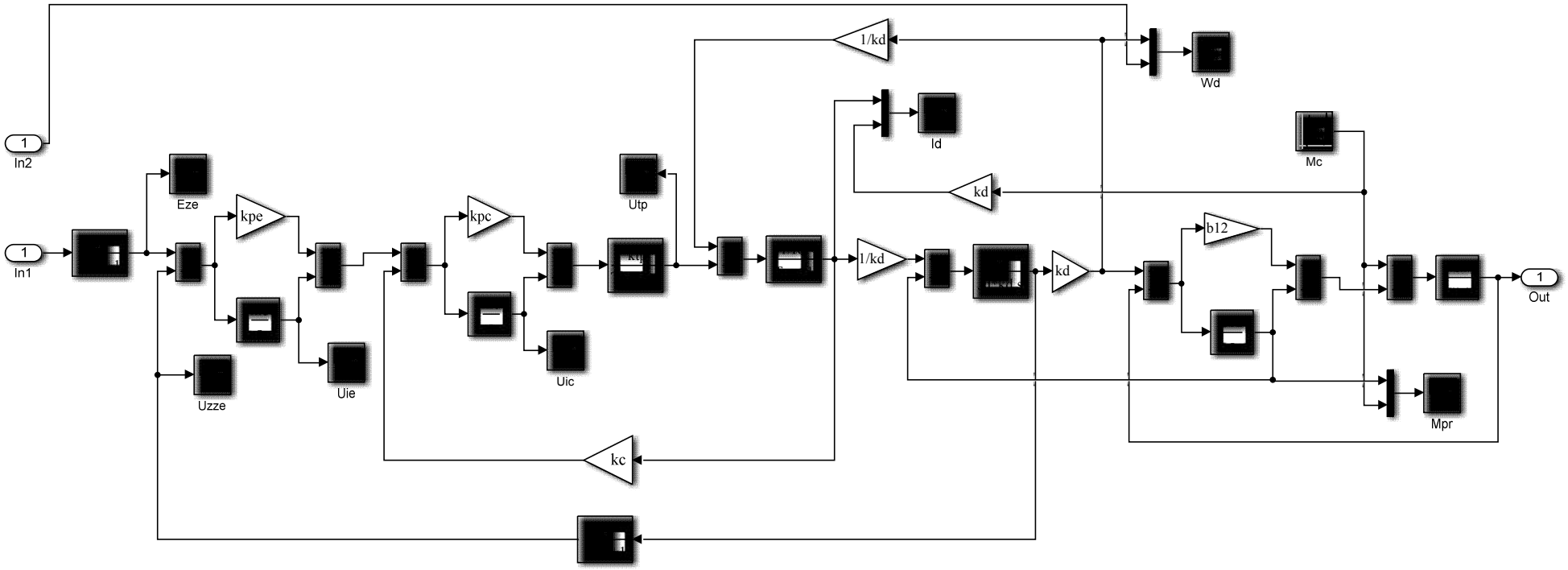


Рисунок 4.9 – Модель об'єкту управління нейрорегулятора

Generate Training Data (генерація навчальних даних). Ця функція запускає процес створення навчальної послідовності для ідентифікації динаміки системи на основі заданих параметрів та моделі об'єкта. Вона дозволяє формувати симульовані дані для навчання, коли реальні виміряні дані недоступні.

Import Data (імпорт даних). Дозволяє завантажити навчальну послідовність із робочої області або зовнішнього файлу для використання у процесі ідентифікації.

Export Data (експорт даних). Надає можливість зберегти згенеровані дані в робочу область Simulink або у файл формату MAT для подальшого використання.

Секція Training Parameters (параметри навчання). Параметри навчання (Training Parameters) визначають, яким чином буде організовано процес навчання моделі системи на основі навчальних даних. Вони забезпечують керування процесом ідентифікації та впливають на точність і ефективність отриманої моделі.

Training Epochs (кількість епох навчання). Параметр Training Epochs задає число ітерацій або повних циклів навчання нейромережі. Кожна епоха передбачає, що модель проходить через весь набір навчальних даних, оновлюючи внутрішні параметри для поліпшення точності відтворення динаміки системи.

Правильне визначення кількості епох є критично важливим: якщо їх занадто мало, модель може залишитися недостатньо навченою і не здатною точно відтворювати поведінку системи; надмірна кількість епох може призвести до перенавчання (overfitting), коли модель запам'ятовує навчальні дані, але погано узагальнює нові випадки. Оптимальна кількість епох залежить від конкретної задачі ідентифікації та характеристик навчальних даних.

Training Function (навчальна функція). Параметр Training Function визначає алгоритм, за допомогою якого модель нейромережі навчається на підготовлених даних та оптимізує свої параметри. Від правильного вибору функції навчання залежить ефективність процесу та точність отриманої моделі.

У MATLAB і пакеті Neural Network Toolbox доступні різні функції навчання, такі як:

- traingd – градієнтний спуск;
- traingdm – градієнтний спуск з моментом;
- trainlm – метод найменших квадратів Левенберга-Марквардта;
- trainbr – байєсівський регулятор.

Кожна з цих функцій має свої особливості та параметри, які можна налаштовувати для досягнення максимальної точності моделі та здатності адекватно відтворювати динаміку системи. Training Function визначає, як нейромережа коригує внутрішні ваги та параметри під час навчання для підвищення якості прогнозу та стабільності моделі.

Use Current Weights (використовувати поточні ваги). Це елемент керування, який дозволяє підтвердити, що під час подальшої ідентифікації та навчання нейронної мережі слід використовувати вже наявні, попередньо сформовані вагові коефіцієнти.

Параметр *Use Current Weights* визначає, чи застосовуватимуться актуальні значення ваг як стартова точка для нового процесу навчання моделі. Він безпосередньо впливає на ініціалізацію мережі:

- Якщо *Use Current Weights* встановлено на «Так» (активовано), тоді мережа починає тренування зі значеннями ваг, які сформувалися під час попередніх навчальних процедур. Тобто навчання продовжує вдосконалення вже наявної моделі.
- Якщо параметр вимкнений («Ні»), навчання розпочинається знову – ваги ініціалізуються випадковим чином або відповідно до обраного алгоритму початкової ініціалізації.

Використання вже наявних ваг доцільне, коли потрібно досягти покращення або точнішої корекції існуючої моделі, а також коли попередні ваги вже певною мірою відображають динаміку об’єкта. Важливо розуміти, що цей параметр визначає стартові умови процесу навчання та може суттєво впливати на кінцеву точність ідентифікації системи.

Use Validation Data (використовувати дані валідації). Цей параметр задає, чи потрібно залучати окрему вибірку даних для етапу валідації під час навчання нейронної мережі. Валідація є важливою процедурою, що дозволяє оцінити здатність моделі узагальнювати інформацію, а не просто запам'ятовувати навчальні приклади.

- Якщо опцію *Use Validation Data* активовано («Так»), навчальний процес використовує додатковий набір даних, який не входить до навчальної вибірки. Це дає змогу оцінити продуктивність моделі на даних, які вона не бачила під час навчання. Такий підхід допомагає виявити перенавчання та визначити рівень узагальнення.
- Якщо параметр вимкнено («Ні»), валідація не проводиться, і результати оцінки моделі ґрунтуються лише на навчальній вибірці. Такий варіант може застосовуватися у випадках, коли немає можливості сформувати додатковий набір даних.

Використання валідаційної вибірки зазвичай рекомендується, оскільки вона забезпечує більш точну оцінку якості моделі та допомагає уникнути деградації її узагальнюючих властивостей.

Use Testing Data (використовувати тестові дані). Це налаштування визначає, чи буде застосовано спеціальний набір тестових даних для підсумкової оцінки роботи моделі після завершення процесу навчання та валідації. Тестові дані є незалежною вибіркою, яку модель ніколи не бачить під час навчання.

- Якщо параметр *Use Testing Data* встановлено на «Так», то після навчання на тренувальних даних і перевірки на валідаційних, модель проходить додаткову оцінку на тестовому наборі. Це дозволяє отримати об'єктивну картину того, наскільки якісно модель зможе працювати з абсолютно новими сигналами та чи здатна вона коректно відтворювати реальну динаміку системи.
- Якщо параметр вимкнено («Ні»), тестування не використовується, і оцінювання моделі обмежується навчальною та, за потреби, валідаційною вибірками.

Використання тестових даних є важливим етапом для перевірки здатності моделі до узагальнення, запобігання перенавчанню та оцінки її практичної придатності.

Процедура Generate Training Data (генерувати навчальні дані). Після вибору цієї опції запускається програмний модуль, який формує навчальну послідовність. Генерація відбувається шляхом подачі на модель керованого об’єкта в середовищі *Simulink* набору випадкових ступінчастих сигналів. У процесі формування навчальних даних на екран виводяться графіки вхідних та вихідних сигналів об’єкта (рис. 4.10), що дозволяє візуально контролювати характер реагування системи.

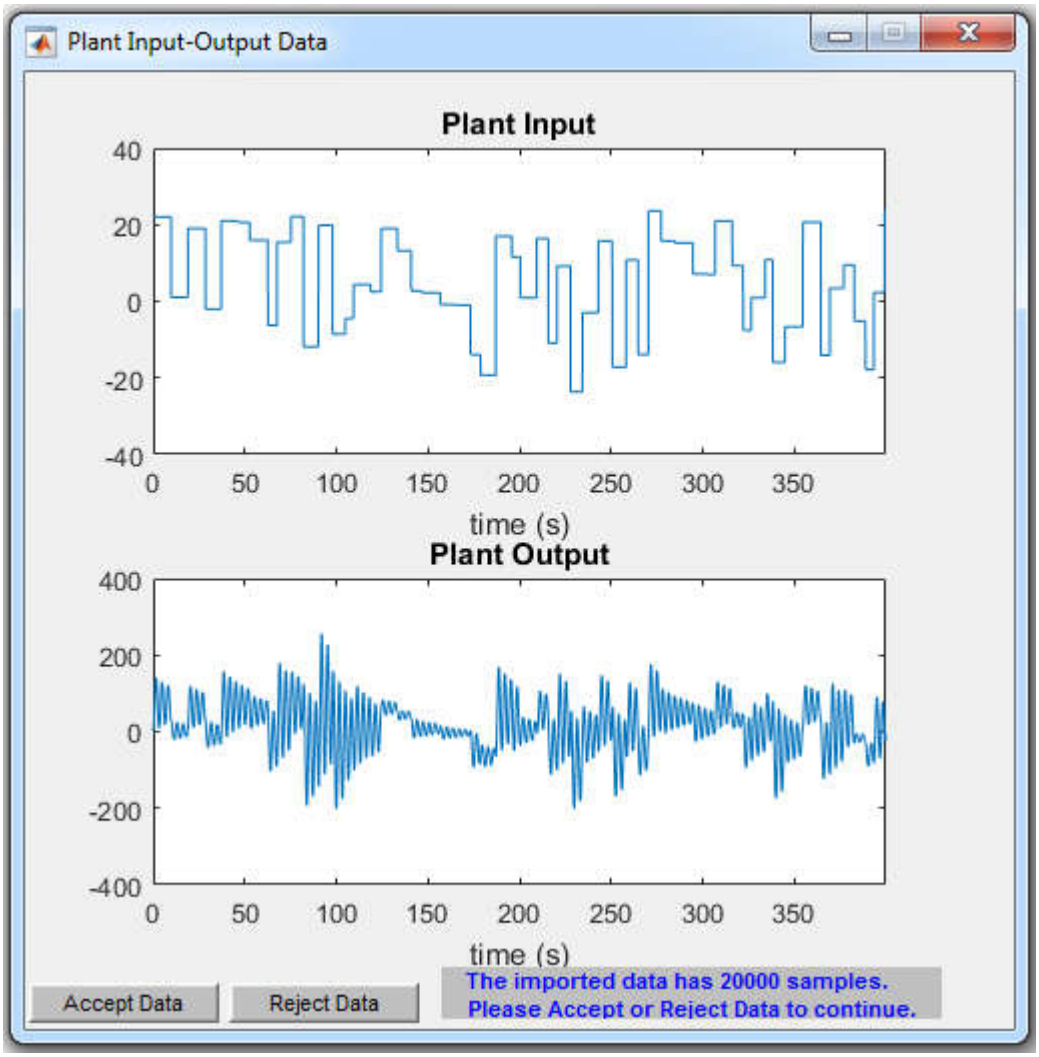


Рисунок 4.10 – Графіки вхідного та вихідного сигналів під час генерації навчальної послідовності

Після завершення генерації користувачу пропонується прийняти отримані дані (*Accept Data*) або відхилити їх (*Reject Data*), якщо вони не відповідають вимогам щодо структури чи якості.

Якщо дані прийняті користувачем, програма автоматично відкриває модифіковане вікно **Plant Identification** (рис. 4.11). У цьому вікні деякі елементи інтерфейсу стають недоступними для редагування, а кнопка **Generate Training Data** змінюється на кнопку **Erase Generate Data**, що надає можливість видалити раніше згенеровані навчальні дані.

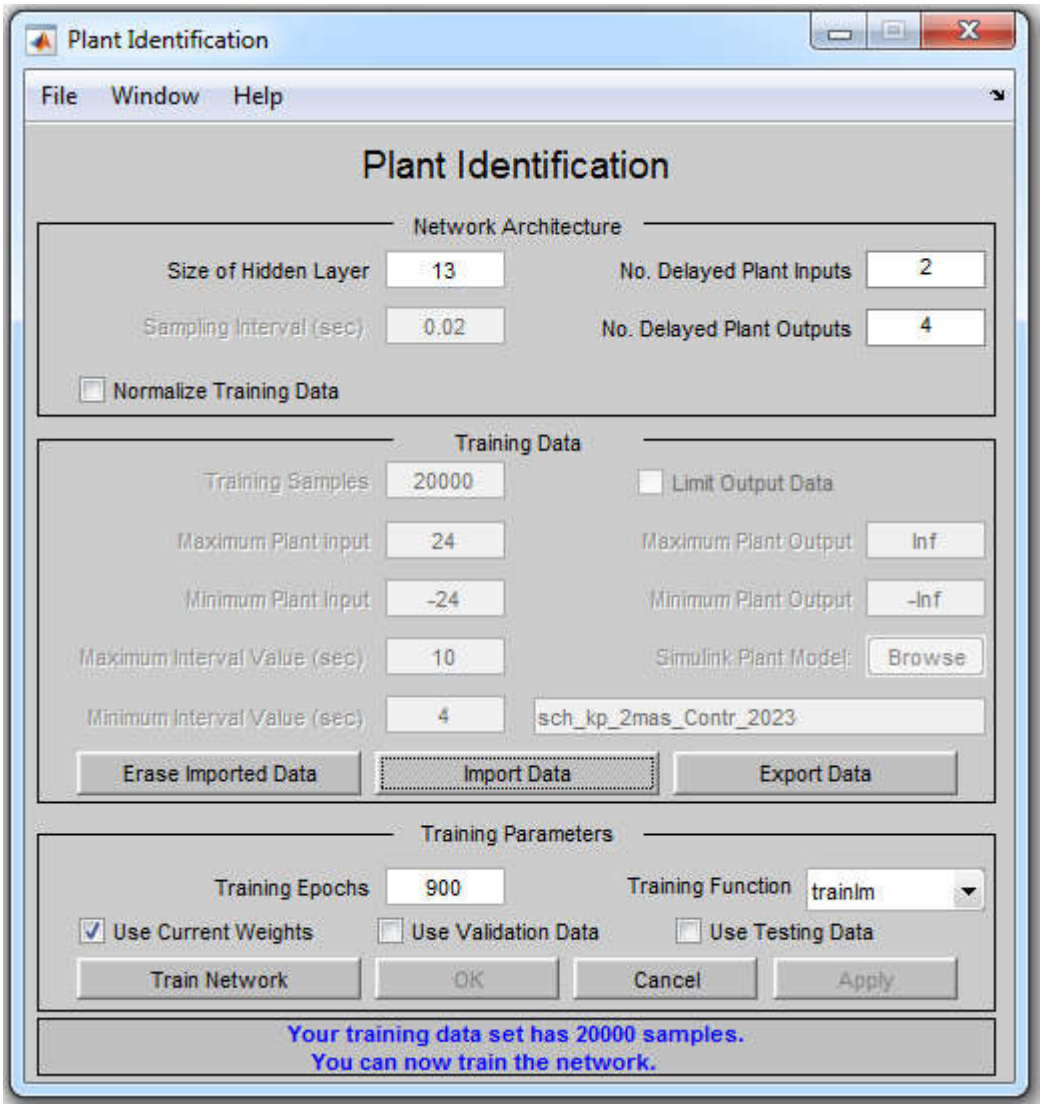


Рисунок 4.11 – Вікно Plant Identification після генерації навчальної послідовності

Натискання кнопки **Train Network** ініціює створення нейронної мережі з прямим розподілом сигналу за допомогою М-функції **newff**. Ця функція не тільки формує мережу з ім'ям **netn**, але й автоматично ініціалізує її ваги та зсуви, готуючи мережу до навчання. Створену нейронну мережу можна відобразити в середовищі Simulink за допомогою оператора **gensim(netn)** (див. рис. 4.12).

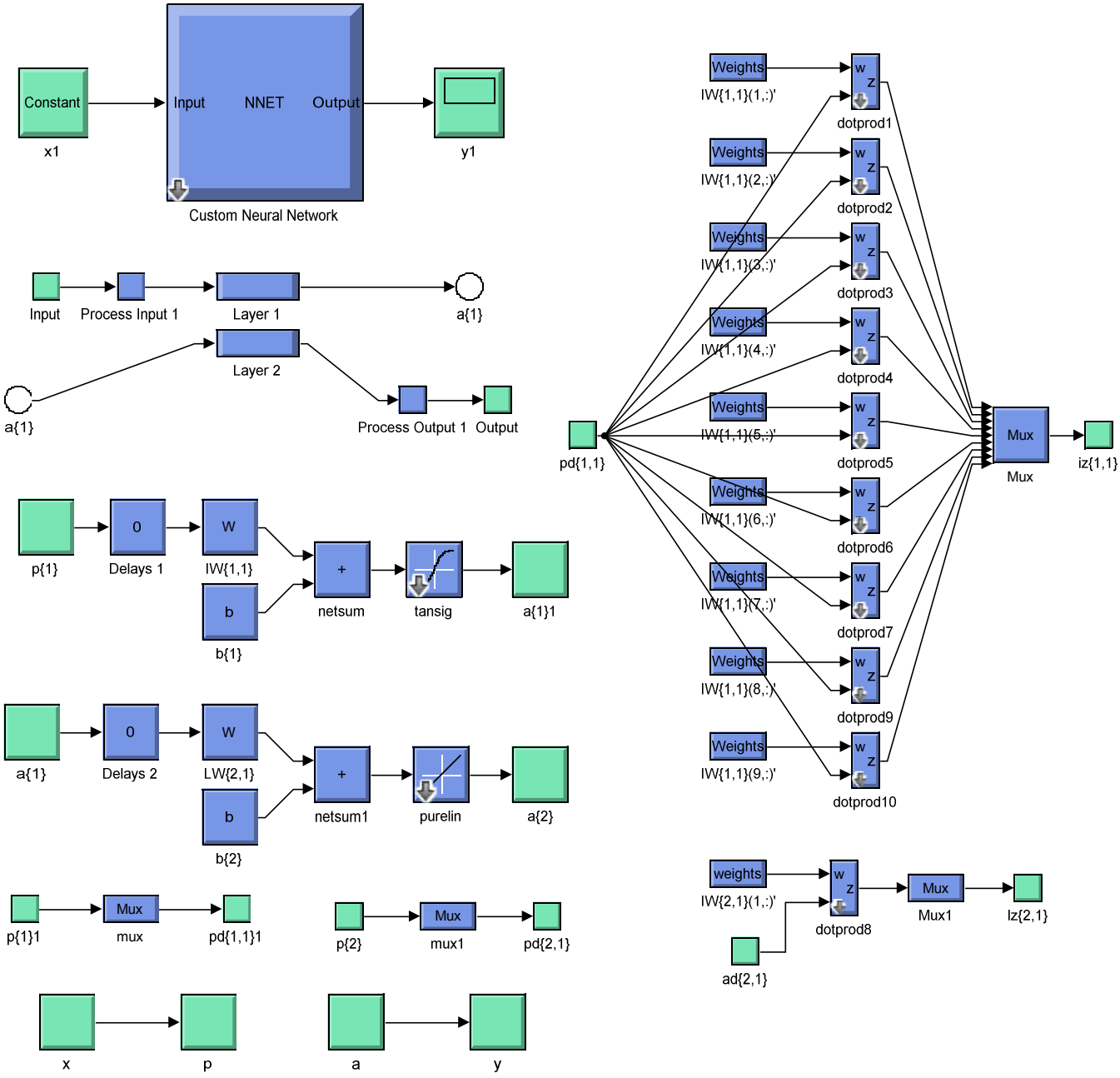


Рисунок 4.12 – Моделі елементів мережі з прямою передачею сигналу, реалізовані в Simulink

Елементи нейронної мережі відповідають заданим параметрам: розмір прихованого шару, кількість затримок на вході та кількість затримок на виході моделі. Кожен наступний елемент мережі стає доступним у новому вікні після подвійного клацання на попередньому елементі. На основі цих елементів у Simulink можна побудувати схему мережі, як показано на рис. 4.13.

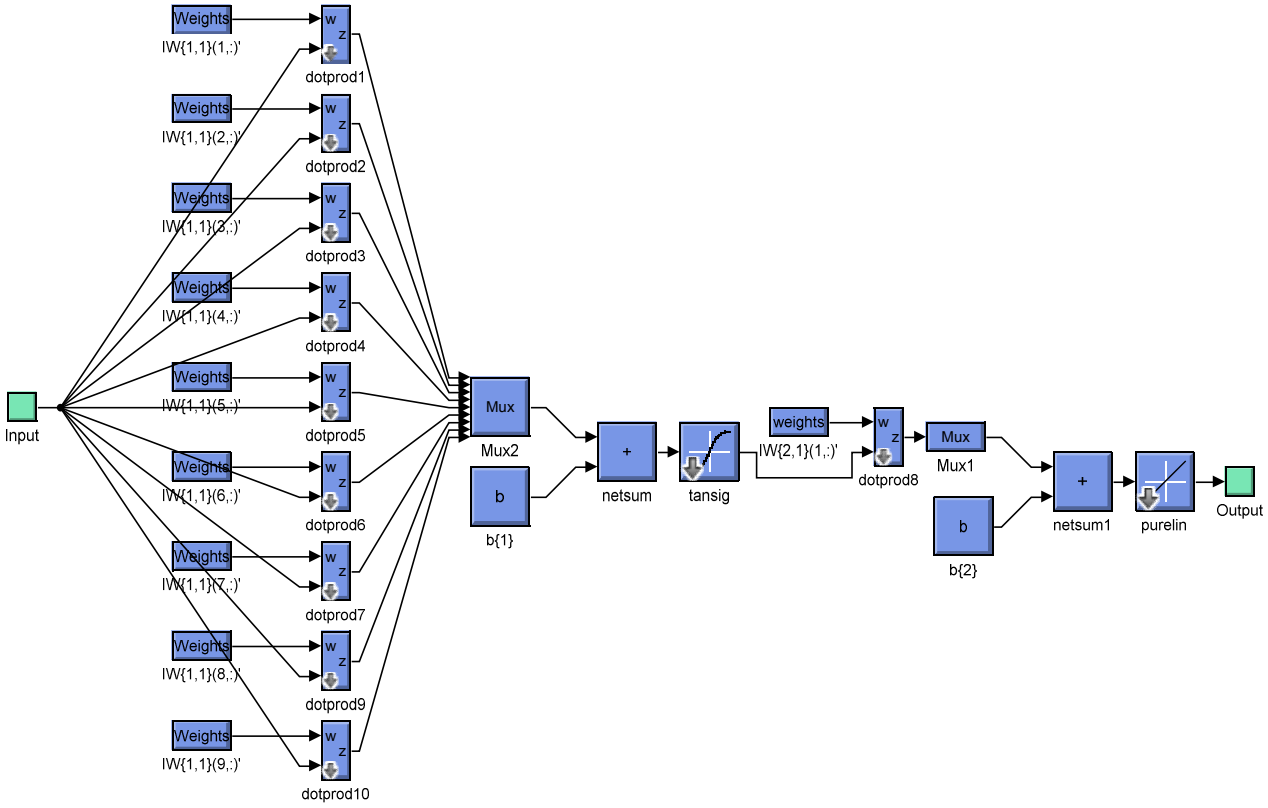


Рисунок 4.13 – Модель статичної мережі з прямою передачею сигналу, побудована в Simulink

Створена мережа є статичною, тобто вона не містить елементів затримки або зворотних зв'язків. Мережа приймає один вектор входу з шістьма елементами, має два шари: перший – прихований, з визначеною кількістю нейронів, другий – вихідний, що складається з одного нейрона. Активаційні функції обрані такі: у прихованому шарі використовується гіперболічний тангенс (**tansig**), а у вихідному шарі – лінійна функція (**purelin**).

Після створення мережі розпочинається процес її навчання. Процес навчання та його результати наочно демонструються через діалогове вікно **Neural**

Network Training (nntraintool), яке показано на рис. 4.14. Це вікно є основним інтерфейсом для навчання нейронних мереж у **Neural Network Toolbox** системи MATLAB. Воно забезпечує потужні можливості для налаштування мережі, візуалізації процесу навчання та моніторингу динаміки зміни параметрів мережі під час тренування.

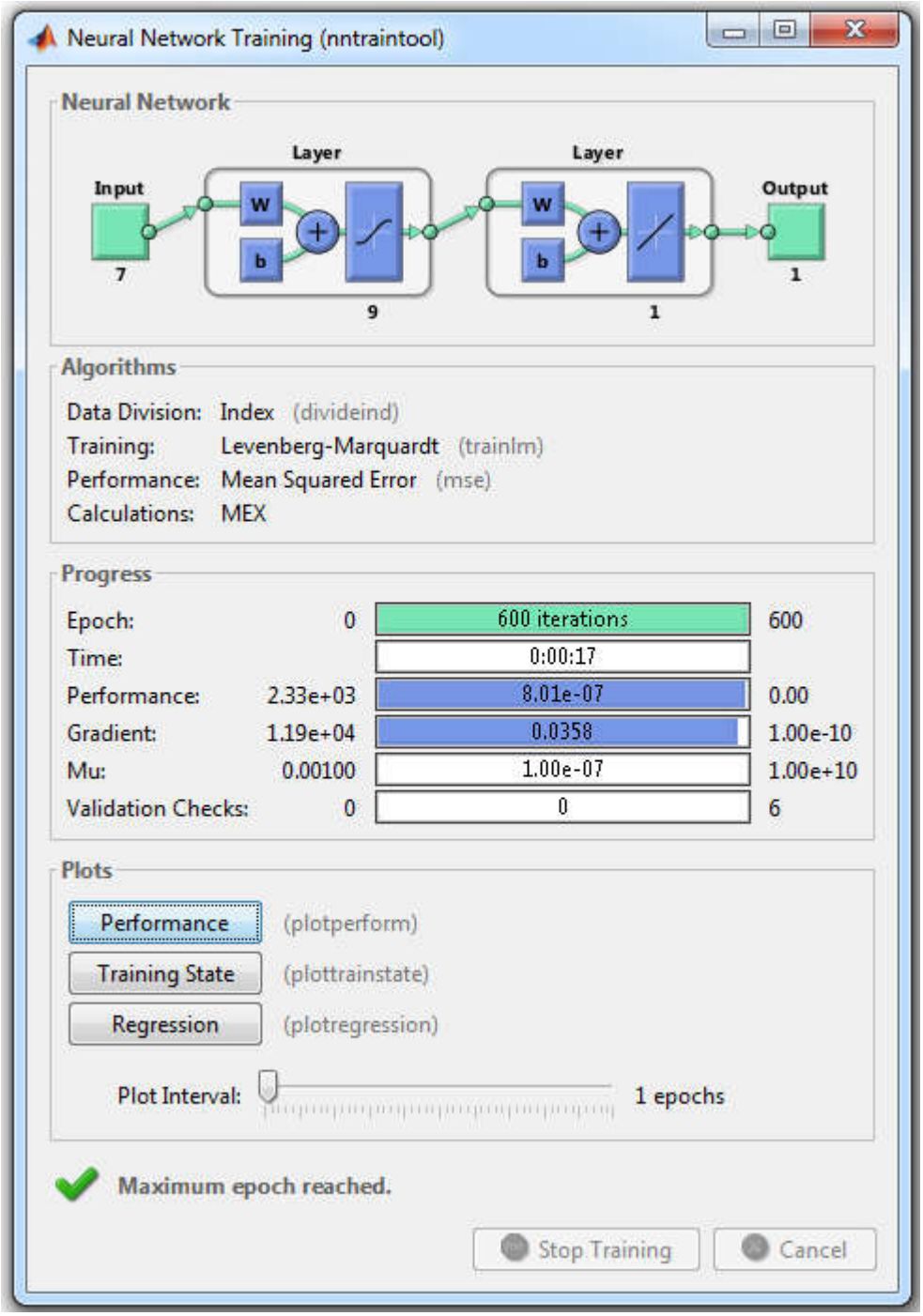


Рисунок 4.14 - Головне вікно для відображення процесу та результатів навчання нейронної мережі

Опис вікна Neural Network Training (nntraintool)

Вікно **Neural Network Training (nntraintool)** у MATLAB складається з декількох функціональних секцій, кожна з яких відповідає за окремий аспект навчання нейронної мережі.

Секція Neural Network (нейронна мережа). У цій секції відображається структура згенерованої мережі, включаючи кількість шарів, нейронів у кожному шарі, а також типи активаційних функцій. Користувач може наочно бачити топологію мережі та взаємозв'язки між шарами, що допомагає оцінити її складність та потенційну здатність до моделювання системи.

Секція Algorithms (алгоритми навчання). Тут відображаються алгоритми, які використовуються для тренування мережі, а також параметри цих алгоритмів. Це дозволяє вибирати відповідний метод оптимізації для конкретного завдання та регулювати швидкість навчання, величину кроку, критерії зупинки та інші параметри, що впливають на процес адаптації ваг і зсувів мережі.

Data Division (поділ даних). Параметр Data Division: Index визначає, які приклади з набору даних будуть використані для навчання, валідації та тестування нейронної мережі. Цей механізм дозволяє контролювати, яка частина даних йде на формування навчальної моделі, а яка використовується для перевірки її здатності до узагальнення. Якщо дані не розподіляються на підмножини, рядок Data Division у вікні nntraintool не відображається.

Training (навчання). Ця секція відповідає за процес оновлення ваг і зсувів нейронної мережі під час тренування. Серед популярних алгоритмів навчання доступні Backpropagation**,** Levenberg-Marquardt**,** Bayesian Regularization та інші.

Вибір Training: Levenberg-Marquardt (trainlm) означає, що мережа буде навчатися за допомогою алгоритму Левенберга-Марквардта. Цей метод поєднує властивості методу найменших квадратів та методу Ньютона для ефективної оптимізації функції втрат. Його основна перевага полягає в швидкій мінімізації функції втрат і стабільному пошуку локального мінімуму через адаптивне оно-

влення вагових коефіцієнтів мережі. Алгоритм добре підходить для складних навчальних задач, де стандартні методи можуть показувати гірші результати.

Performance: Mean Squared Error (MSE). Параметр відображає показники точності навчання нейронної мережі на основі обраної функції втрат та даних для валідації. Цей показник дає змогу оцінити, наскільки ефективно мережа відтворює динаміку системи під час навчання, а також визначає момент, коли досягнуто оптимальної точності та процес тренування можна завершити. Моніторинг Performance допомагає уникнути перенавчання та забезпечує належну здатність моделі до узагальнення на нових даних.

Performance: Mean Squared Error (MSE) визначає використання середньоквадратичної помилки як критерію оцінки точності нейронної мережі під час її навчання. Середньоквадратична помилка є однією з найпоширеніших метрик у задачах регресії та моделювання. Вона розраховується як середнє значення квадратів різниці між прогнозованими виходами моделі (нейронної мережі) та відомими або фактичними значеннями з навчального набору даних.

Математично MSE можна записати так:

$$MSE = \frac{\sum_{i=1}^{N} (y_{actual_i} - y_{predicted_i})^2}{N}$$

де N – кількість прикладів у наборі даних, y_{actual_i} – фактичне значення виходу, а $y_{predicted_i}$ – прогнозоване значення, яке видає нейронна мережа.

MSE показує середнє квадратичне відхилення між фактичними та передбаченими результатами. Чим менше значення MSE, тим точніше модель повторює поведінку системи. Під час навчання мережі оптимізаційний алгоритм прагне зменшити MSE, тобто підбирати ваги мережі так, щоб прогнозовані значення максимально відповідали реальним даним.

Calculations (обчислення). Секція Calculations відповідає за контроль кількості обчислень, виконаних під час процесу навчання. Вона важлива для оці-

нки продуктивності та часу, необхідного для завершення тренування нейронної мережі.

Параметр Calculations: MEX визначає, чи слід використовувати MEX-функції для прискорення обчислень у MATLAB. MEX (MATLAB Executable) дозволяє запускати оптимізований код, написаний на C, C++ або Fortran, безпосередньо у MATLAB, що забезпечує значно більшу швидкість виконання порівняно зі стандартним інтерпретованим кодом MATLAB.

У контексті навчання нейронних мереж використання MEX-функцій дозволяє швидше виконувати складні математичні обчислення, що особливо корисно при роботі з великими наборами даних або складними моделями. Завдяки MEX-файлам тренування мережі стає ефективнішим, зменшується час очікування результатів і підвищується продуктивність процесу.

Секція Progress (прогрес навчання). У цій секції відображається поточний стан навчання нейронної мережі в реальному часі. Вона надає користувачу можливість спостерігати за ходом навчання та оцінювати ефективність процесу. Інформація, що міститься у секції, допомагає контролювати навчання моделі, виявляти проблеми на ранніх стадіях та своєчасно вносити необхідні коригування для підвищення точності та стабільності моделі.

Epoch (епоха). Параметр Epoch означає повний прохід усіх прикладів навчального набору даних через нейронну мережу. Під час кожної епохи мережа отримує всі вхідні дані один раз, після чого відбувається оновлення її вагових коефіцієнтів та зсувів (biases) відповідно до обчислених помилок. Кількість епох визначає, скільки разів модель "пройде" через дані, що дозволяє поступово покращувати її здатність відтворювати динаміку системи. Відстеження епох дозволяє оцінювати, наскільки ефективно мережа навчається, і приймати рішення про зупинку або продовження тренування.

Time (час). Параметр Time показує фактичний час, витрачений на процес навчання нейронної мережі. Це допомагає контролювати тривалість навчання,

планувати ресурси та оцінювати продуктивність навчальної сесії, особливо при роботі з великими наборами даних або складними моделями.

Performance (ефективність/якість навчання). Показник Performance відображає, наскільки успішно нейронна мережа навчається на даних, використовуючи обрану функцію втрат (loss function). Для задач регресії найчастіше застосовується середньоквадратична помилка (Mean Squared Error, MSE)**,** яка обчислює середнє значення квадратів різниці між прогнозованими виходами та реальними значеннями цільової змінної. Чим менше значення MSE, тим точніше мережа відтворює поведінку системи. Показник Performance дозволяє оцінити ефективність навчання на тренувальних та валідаційних даних і допомагає приймати рішення про необхідність коригування параметрів навчання.

Gradient (градієнт). Параметр Gradient демонструє вектор градієнта функції втрат відносно всіх параметрів нейронної мережі, тобто ваг і зсувів. Градієнт показує напрямок і величину змін, які слід застосувати до параметрів мережі, щоб мінімізувати функцію втрат. Під час навчання мережі використовується метод зворотного поширення помилки (backpropagation)**,** і градієнт є ключовим елементом цього процесу. Він дозволяє алгоритму навчання ефективно коригувати ваги та зсуви, забезпечуючи поступове зменшення помилки і підвищення точності моделі.

Таким чином, секція Progress у nntraintool дозволяє детально контролювати процес навчання нейронної мережі, оцінювати якість навчання на кожному етапі та своєчасно реагувати на будь-які відхилення або проблеми, що виникають під час тренування.

Mu (тау-параметр). Параметр Mu відноситься до так званого тау-параметра, який застосовується в алгоритмах оптимізації для регулювання швидкості навчання нейронної мережі. Він особливо актуальний при використанні адаптивних методів оптимізації, таких як зворотне поширення помилки (backpropagation) або алгоритми на кшталт BFGS (Broyden-Fletcher-Goldfarb-Shanno)**.** Основна функція Mu полягає у корекції величини кроку навчання відповідно до поточної динаміки процесу оптимізації. Це дозволяє алгоритму ефе-

ктивно знаходити мінімум функції втрат і забезпечує стабільну збіжність навіть при складних поверхнях втрат або великих градієнтах.

Значення Mu зазвичай обчислюється автоматично алгоритмом на кожній ітерації навчання та може змінюватися в процесі, підлаштовуючись під поведінку мережі. Від правильного регулювання цього параметра залежить швидкість та стабільність навчання, а також здатність мережі до ефективного узагальнення даних.

Validation Checks (перевірки на валідаційному наборі). Параметр Validation Checks визначає кількість перевірок точності або якості нейронної мережі під час процесу навчання за допомогою окремого набору даних, який не бере участі у безпосередньому навчанні – так званого validation dataset. Основна мета цих перевірок – контроль за процесом навчання і запобігання перенавчанню (overfitting), коли модель «запам'ятовує» навчальні дані і втрачає здатність правильно реагувати на нові, невідомі приклади.

Наприклад, якщо встановлено 6 перевірок, нейронна мережа після кожних шести епох (де епоха – це повний прохід через весь навчальний набір даних) перевірятиме свою продуктивність на валідаційному наборі. Якщо точність моделі на цих даних перестає покращуватися або починає погіршуватися, це сигнал для зупинки навчання, оскільки подальші ітерації можуть призвести до перенавчання.

Таким чином, Validation Checks дозволяє автоматично відслідковувати ефективність мережі на даних, які вона раніше не бачила, і забезпечує своєчасне завершення навчання для досягнення оптимального балансу між підгонкою під тренувальні дані та здатністю узагальнювати на нові дані.

Секція Plots (графіки). Ця секція призначена для візуалізації ходу навчання нейронної мережі і дозволяє відслідковувати зміни її продуктивності у реальному часі. Візуальні графіки допомагають аналізувати ефективність навчання, виявляти проблеми, такі як перенавчання, та оцінювати динаміку зменшення помилки на різних етапах тренування.

Кнопка **Performance** відкриває вікно **Neural Network Training Performance** (рис. 4.15), в якому представлені графіки середньоквадратичної помилки (**Mean Squared Error, MSE**) для навчального, валідаційного та тестового наборів даних. Ці графіки демонструють, як змінюється точність мережі під час кожної ітерації навчання.

Графіки **MSE** дозволяють відстежувати, чи відбувається зменшення помилки на всіх наборах даних і чи немає ознак перенавчання, коли помилка на валідаційному та тестовому наборі починає збільшуватися. Основна мета – досягти мінімальної помилки на всіх наборах даних для отримання максимально точної та узагальнювальної моделі.

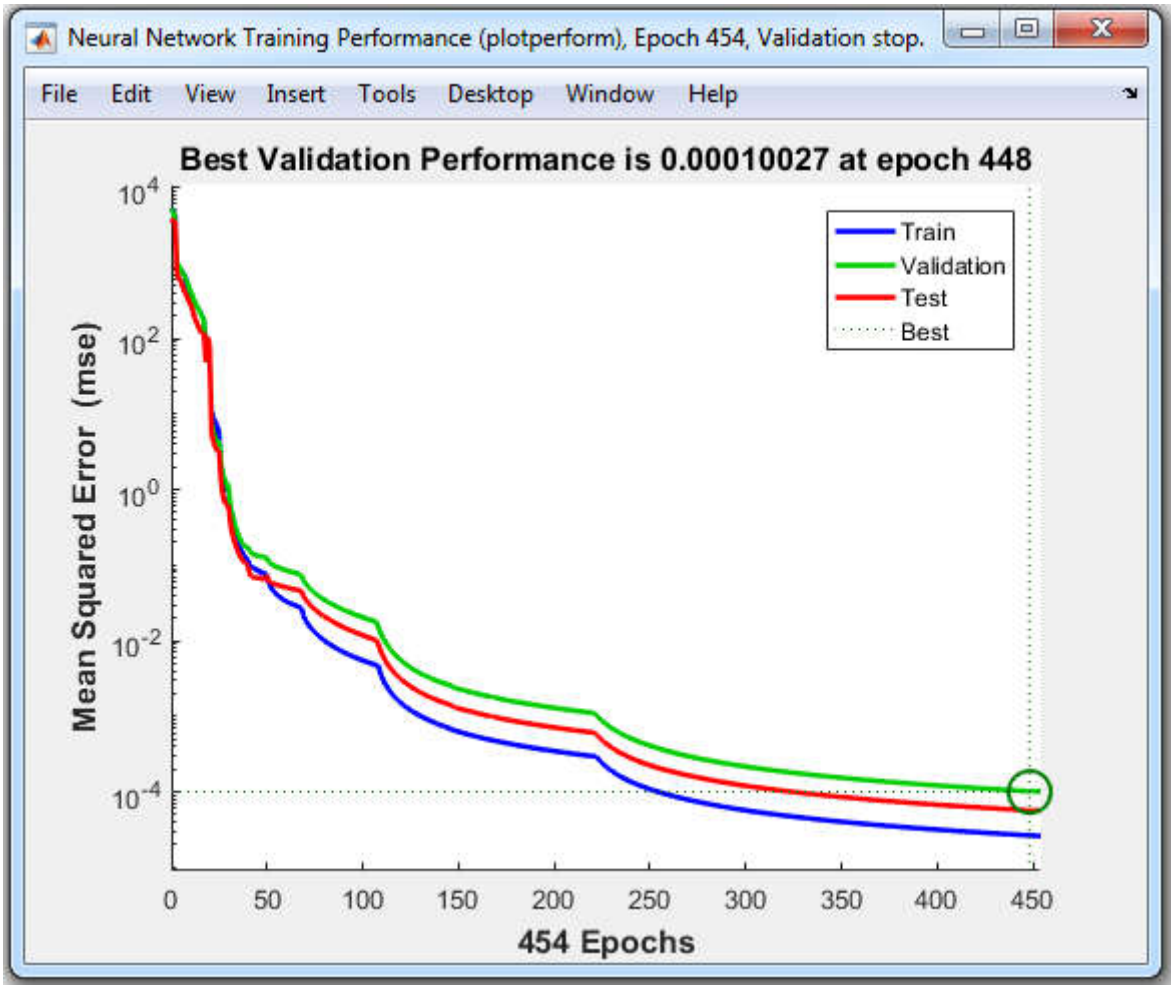


Рисунок 4.15 – Вікно контролю процесу навчання

Кнопка **Training State** відкриває вікно **Neural Network Training State** (рис. 4.16), яке відображає стан навчання нейронної мережі на конкретний момент часу. У цьому вікні можна переглядати актуальні значення таких параметрів, як кількість епох, поточна середньоквадратична помилка, градієнти, тау-параметр (Mu) та інші показники. Це дозволяє детально відстежувати процес навчання та приймати рішення про його коригування, наприклад, про зупинку навчання у разі виникнення перенавчання або про необхідність зміни налаштувань алгоритму.

Таким чином, секція **Plots** є ключовим інструментом для аналізу навчального процесу, що забезпечує як візуальний контроль, так і можливість оперативно реагувати на будь-які відхилення у процесі тренування нейронної мережі.

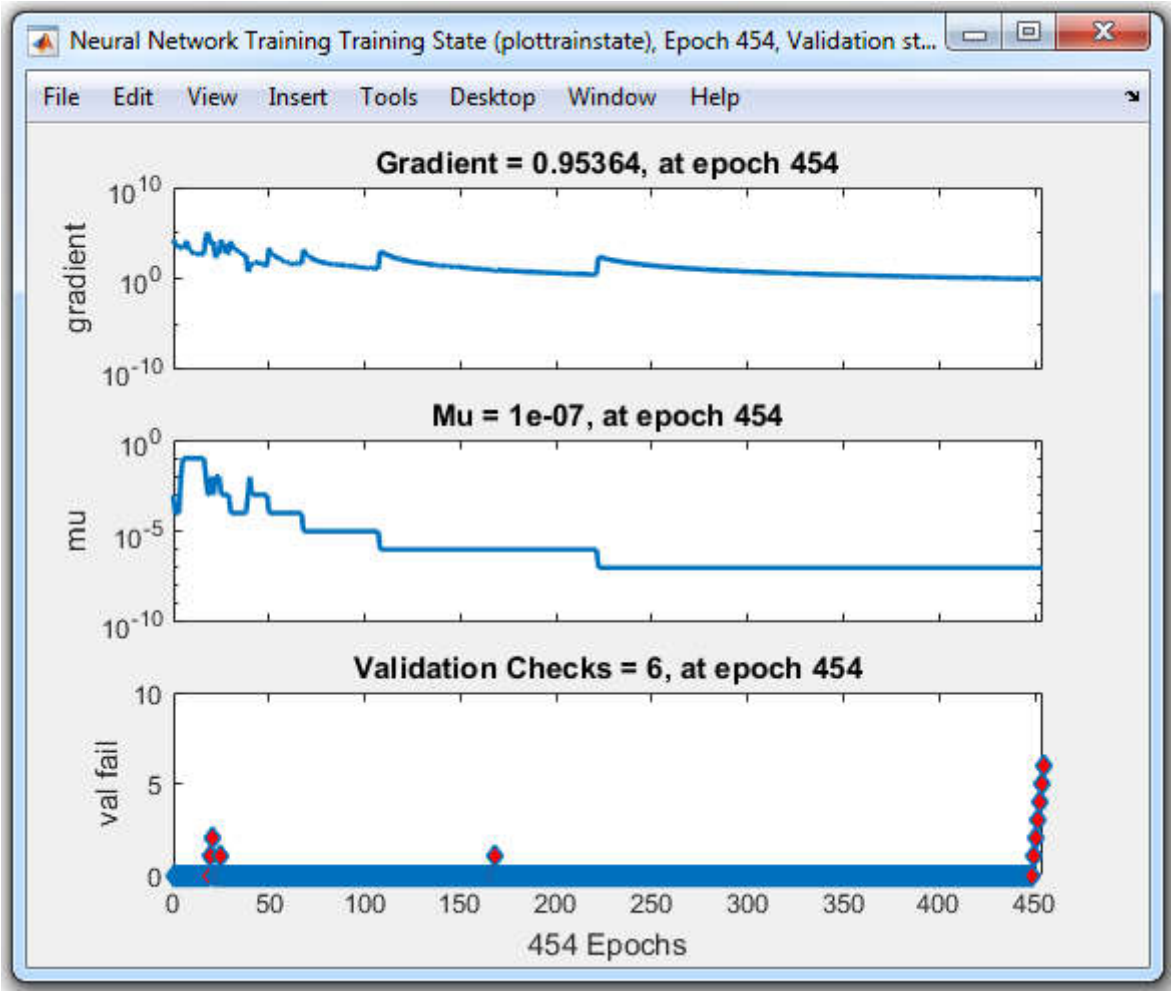


Рисунок 4.16 - Вікно Neural Network Training State

Опис вікна Neural Network Training State

У вікні **Neural Network Training State** відображаються основні графіки, які дозволяють відстежувати динаміку навчання нейронної мережі в реальному часі. Це вікно дає змогу детально контролювати процес тренування та своєчасно реагувати на можливі проблеми.

Графік Gradient (градієнт). Цей графік демонструє зміну величини градієнта функції втрат під час навчання мережі. У контексті нейронних мереж градієнт є вектором, який вказує напрямок та швидкість найшвидшого зростання функції помилки. Використовуючи цю інформацію, оптимізаційний алгоритм коригує вагові коефіцієнти мережі в протилежному напрямку, щоб мінімізувати значення функції втрат.

На графіку Gradient можна побачити, як змінюється величина градієнта від початкових ітерацій до моменту, коли мережа наближається до оптимальних вагових коефіцієнтів. Спочатку градієнти можуть бути значними, але з часом вони зазвичай зменшуються, що свідчить про поступове зближення мережі до стабільного стану.

Моніторинг цього графіка допомагає виявляти проблеми gradient vanishing або gradient exploding та контролювати збіжність навчання. Надто великі або малі значення градієнта можуть сигналізувати про необхідність коригування параметрів навчання, таких як швидкість навчання або структура мережі.

Графік Mu (параметр навчання). Графік mu відображає зміну значення тау-параметра, який регулює швидкість навчання та рівень регуляризації мережі. У процесі тренування оптимізаційні алгоритми (наприклад, стохастичний градієнтний спуск або модифікації методу Бройдена-Флетчера-Гольдфарба-Шанно) використовують mu для адаптації швидкості корекції вагових коефіцієнтів.

Зміни параметра mu показують, як адаптується швидкість навчання до поточного стану мережі. Наприклад, збільшення mu може прискорити збіж-

ність, але водночас збільшити ризик нестабільності, тоді як зменшення mu може свідчити про уповільнення навчання або необхідність перегляду налаштувань оптимізатора. Регулярний контроль графіка mu дозволяє підтримувати баланс між швидкістю збіжності та стабільністю процесу навчання.

Графік Validation Failure (val fail). Цей графік демонструє зміну показника невдач на наборі валідаційних даних протягом навчання. Основне призначення валідаційного набору – оцінити здатність мережі узагальнювати знання на дані, які не використовувалися під час тренування, і запобігти перенавчанню.

Якщо кількість невдач на валідації зростає, це може вказувати на overfitting, тобто на надмірне підлаштування мережі під навчальні дані та втрату здатності правильно передбачати нові вхідні сигнали. У таких випадках доцільно застосувати ранню зупинку навчання, змінити гіперпараметри або додатково скоригувати структуру мережі. Мінімізація цього показника є однією з ключових цілей тренування, адже вона прямо впливає на точність та стабільність моделі в реальних умовах.

Таким чином, Neural Network Training State забезпечує комплексний моніторинг навчального процесу, дозволяючи спостерігати за градієнтами, швидкістю навчання та поведінкою мережі на валідаційних даних. Ця інформація є критично важливою для контролю якості навчання та своєчасного коригування параметрів для досягнення оптимальної моделі.

Продовження опису вікна Neural Network Training (nntraintool)

Кнопка **Regression (регресія)** призначена для налаштування параметрів навчання нейронної мережі у задачах регресії. Регресійне завдання – це один із типів задач машинного навчання, мета якого полягає у передбаченні числових значень або кількісних показників на основі вхідних даних. У таких задачах модель намагається відтворити функціональну залежність між вхідними ознаками (параметрами) та вихідними значеннями, які називаються цільовими або відгуком.

При натисканні на кнопку **Regression** відкривається вікно **Neural Network Training Regression** (рис. 4.17), яке надає графічне та числове представлення продуктивності моделі під час навчання. У цьому вікні користувач може спостерігати детальну інформацію щодо точності прогнозів мережі на різних наборах даних.

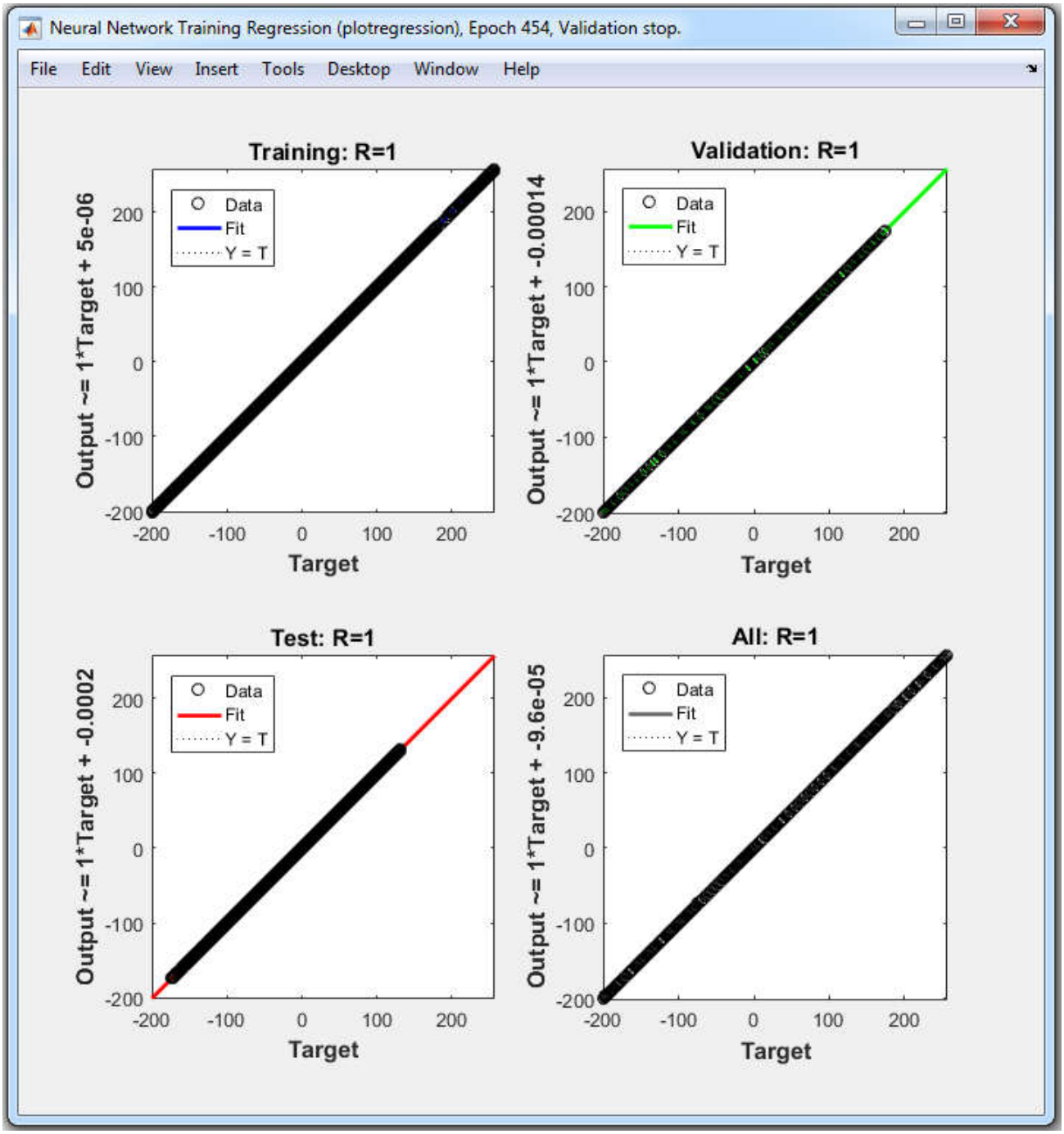


Рисунок 4.17 - Вікно Neural Network Training Regression

Опис секцій вікна Neural Neural Network Training Regression

Training (навчання). Ця секція відображає, наскільки добре нейронна мережа відтворює дані навчального набору, тобто дані, на яких модель проходила тренування. Графіки показують відповідність прогнозованих значень фактичним даним, що дозволяє оцінити ефективність навчання та ступінь узгодження мережі з вихідними даними.

Графіки Validation (валідація). Валідаційні графіки демонструють точність роботи моделі на окремому наборі даних, який не використовувався під час навчання. Валідація дозволяє оцінити, наскільки добре модель узагальнює знання на нові дані, і служить для своєчасного виявлення ознак перенавчання (overfitting).

Графіки Test (тестування). Тестові графіки включають незалежний набір даних, який не був задіяний ні під час навчання, ні під час валідації. Вони забезпечують об’єктивну оцінку продуктивності нейронної мережі на абсолютно нових даних. Аналіз графіків Test дозволяє перевірити, наскільки модель здатна передбачати результати для непередбачених ситуацій.

Графіки All (всі дані разом). Ця секція об’єднує результати прогнозування для всіх трьох наборів даних: Training, Validation та Test. Графіки **All** показують одночасно прогнозовані та фактичні значення для кожного набору, що дозволяє наочно порівнювати продуктивність мережі на різних етапах навчання. Такий підхід допомагає швидко визначити, чи існують розбіжності між поведінкою моделі на навчальних, перевірочних і тестових даних, та робить оцінку її стабільності більш наочною.

Таким чином, вікно **Neural Network Training Regression** забезпечує комплексний аналіз регресійних результатів моделі, дозволяє відстежувати точність прогнозів на всіх основних наборах даних та вчасно коригувати параметри нейронної мережі для досягнення оптимальної продуктивності.

Графіки Data та Fit у вікні Neural Network Training Regression. У вікні Neural Network Training Regression графіки Data для наборів Training, Validation,

Test та All демонструють, наскільки прогнозовані значення моделі нейронної мережі узгоджуються з фактичними даними для кожного з наборів. На горизонтальній осі відкладаються фактичні значення з відповідного набору даних, тоді як на вертикальній осі – прогнозовані значення, отримані від моделі. Кожна точка на графіку відображає відповідність прогнозованого і справжнього значення: координата по осі X відповідає бажаному значенню, а координата по осі Y – вихідному значенню мережі. У середовищі MATLAB точки позначаються кружечками ( ).

Якщо всі точки розташовані близько до діагональної лінії $Y = T$, це свідчить про високу точність моделі. Велика розсіюваність точок може вказувати на наявність шуму в даних або недостатню адекватність моделі. Графіки Data дозволяють наочно оцінити відповідність моделі даним на різних наборах і виявити можливі аномалії чи області для поліпшення моделі.

Графіки **Fit** представляють результати лінійної регресії, що демонструють прогнозовані значення, які генерує нейронна мережа на основі вхідних даних. Вони відображають, наскільки модель узгоджується з фактичними значеннями, і служать для візуальної оцінки точності прогнозів. Якщо лінія Fit проходить близько до більшості точок, це вказує на високу продуктивність моделі. Відхилення лінії від фактичних значень може сигналізувати про потребу в додатковій оптимізації або корекції гіперпараметрів нейронної мережі.

У вікні Neural Network Training Regression також відображаються коефіцієнти кореляції та рівняння лінійної регресії для кожного набору даних (Training, Validation, Test та All). Вони використовуються для кількісної оцінки точності моделі.

Коефіцієнт кореляції R визначає ступінь лінійної залежності між прогнозованими та фактичними значеннями:

- R = 1 – ідеальна лінійна залежність, модель точно передбачає всі значення;
- R = 0 – відсутність лінійної залежності, модель не здатна передбачити дані;

- R = -1 – повна зворотна лінійна залежність, модель передбачає значення у протилежному напрямку.

Коефіцієнт кореляції, близький до 1, свідчить про високу точність регресійної моделі, тоді як значення близько до 0 або від'ємне сигналізує про низьку точність прогнозування. Використання R разом із графіками Data та Fit дозволяє комплексно оцінити ефективність моделі нейронної мережі на всіх етапах навчання та перевірки, а також визначити можливі напрями для її вдосконалення.

Продовження опису вікна Neural Network Training (nntraintool)

Plot Interval (інтервал графіка). Параметр Plot Interval визначає, як часто під час навчання нейронної мережі для задач регресії оновлюються графіки та відображаються проміжні результати. Іншими словами, він встановлює кількість ітерацій або епох, після яких відбувається оновлення інформації на графіках.

Правильне налаштування Plot Interval дозволяє ефективно контролювати обсяг відображуваної інформації під час навчання, особливо коли кількість епох велика. Менше значення Plot Interval забезпечує частіше оновлення графіків, що дає змогу уважніше аналізувати процес навчання, але одночасно збільшує обсяг візуальної інформації. Більше значення, навпаки, зменшує частоту оновлення, спрощуючи відстеження загальної тенденції навчання без перевантаження графіків.

Налаштування цього параметра є важливою частиною контролю процесу навчання, оскільки дозволяє оцінювати вплив змін параметрів мережі на результати моделі в режимі реального часу.

Кнопки Stop Training і Cancel. У вікні **Neural Network Training (nntraintool)** середовища MATLAB присутні дві кнопки для управління процесом навчання, які мають різні функції:

- **Stop Training (зупинка навчання)** – негайно припиняє процес навчання нейронної мережі без можливості його продовження. Ця кнопка викори-

стовується, коли навчання виявляється занадто тривалим, не дає задовільних результатів або потрібно терміново припинити тренування.

- **Cancel (скасувати)** – дозволяє зупинити навчання, але при цьому зазвичай передбачає підтвердження дії та збереження поточного стану мережі. Використання кнопки Cancel дає змогу зупинити навчання після перегляду проміжних результатів, внести необхідні зміни до параметрів та при необхідності продовжити навчання пізніше.

Після завершення навчання результати відображаються у вікні **Training Data for NN Predictive Control**, на якому показані графіки фактичних та прогнозованих значень, що дозволяє оцінити точність моделі та якість її навчання (рис. 4.18).

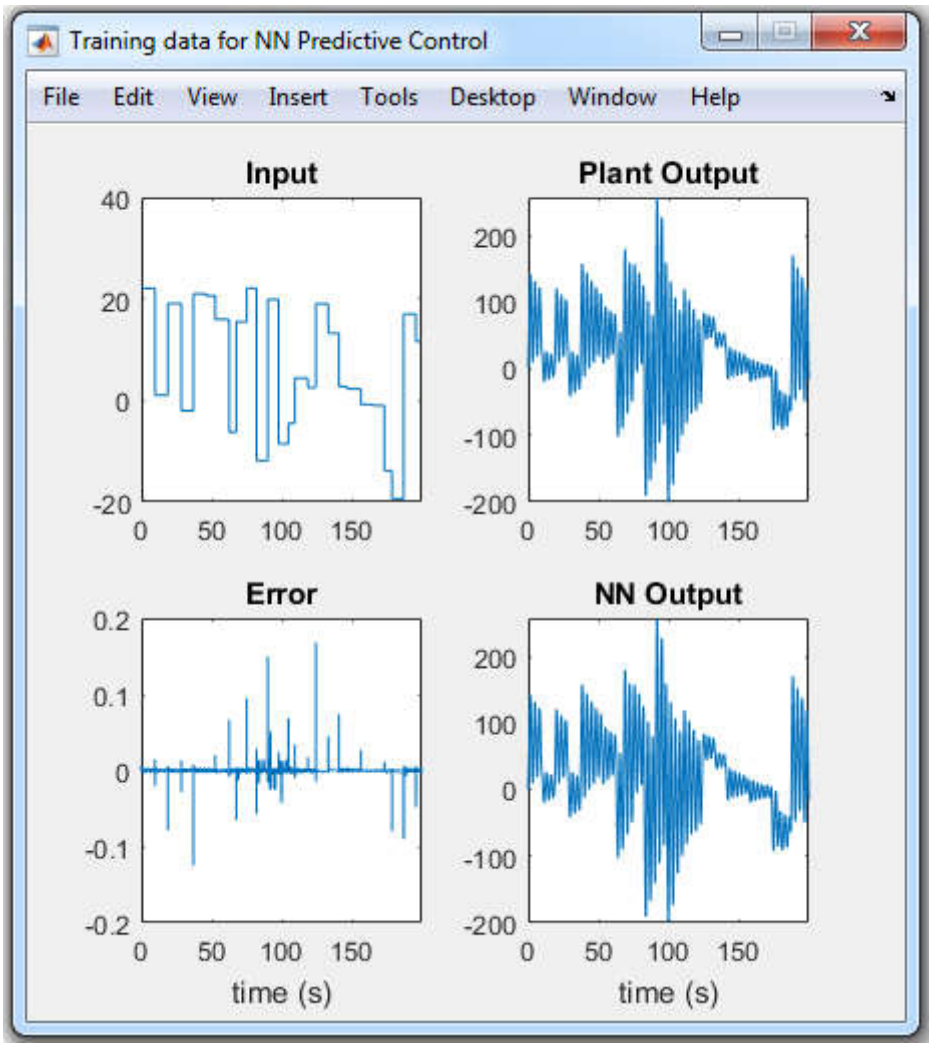


Рисунок 4.18 – Графічне представлення результатів навчання нейронної мережі

Опис вікна Training Data for NN Predictive Control

У вікні **Training Data for NN Predictive Controller** відображаються графіки, що ілюструють процес навчання нейронної мережі та взаємодію моделі з системою:

- **Input (вхідні сигнали)** – графік Input показує вхідні сигнали, які подаються на систему та нейронну мережу під час навчання. Ці сигнали стимулюють модель для генерації відповідей і дозволяють досліджувати її реакцію на різні умови.
- **Plant Output (вихід системи)** – графік Plant Output демонструє фактичні вихідні дані системи (Plant) під час навчання. Він відображає реальну динаміку системи у відповідь на вхідні сигнали, що дозволяє порівняти її поведінку з прогнозами моделі.
- **NN Output (вихід нейромережі)** – графік NN Output відображає прогнозовані вихідні сигнали, згенеровані нейронною мережею під час навчання. Ці дані ілюструють здатність моделі відтворювати поведінку системи на основі поданих вхідних сигналів.
- **Error (помилка)** – графік Error показує різницю між прогнозованими значеннями нейронної мережі (NN Output) та фактичними виходами системи (Plant Output). Цей показник відображає точність моделі та дозволяє оцінити, наскільки адекватно мережа відтворює динаміку системи.

Аналогічні графіки відображаються у вікнах **Validation Data for NN Predictive Control** та **Testing Data for NN Predictive Control (**рис. 4.19, 4.20), які ілюструють результати перевірки моделі на контрольній та тестовій множинах даних. Ці графіки дозволяють оцінити здатність нейронної мережі узагальнювати знання, отримані під час навчання, на нові дані.

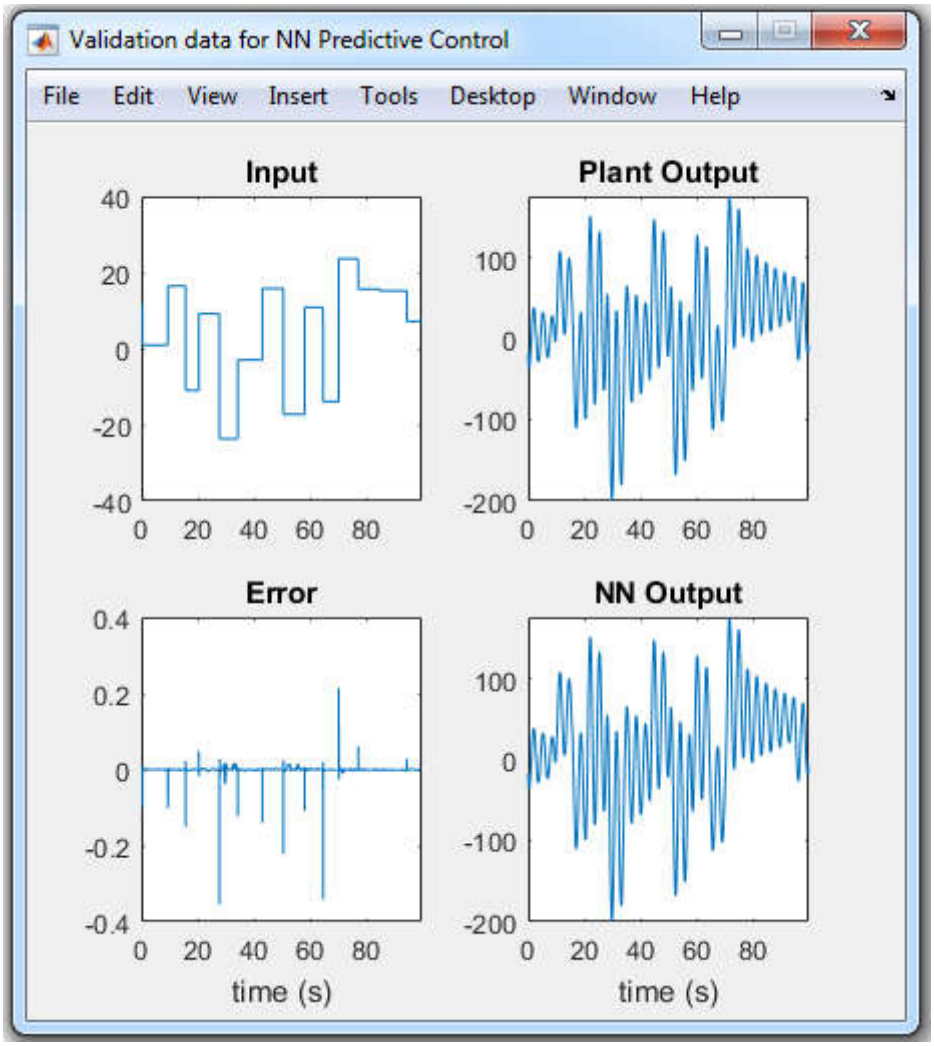


Рисунок 4.19 – Візуалізація результатів перевірки нейронної мережі на контрольній множині

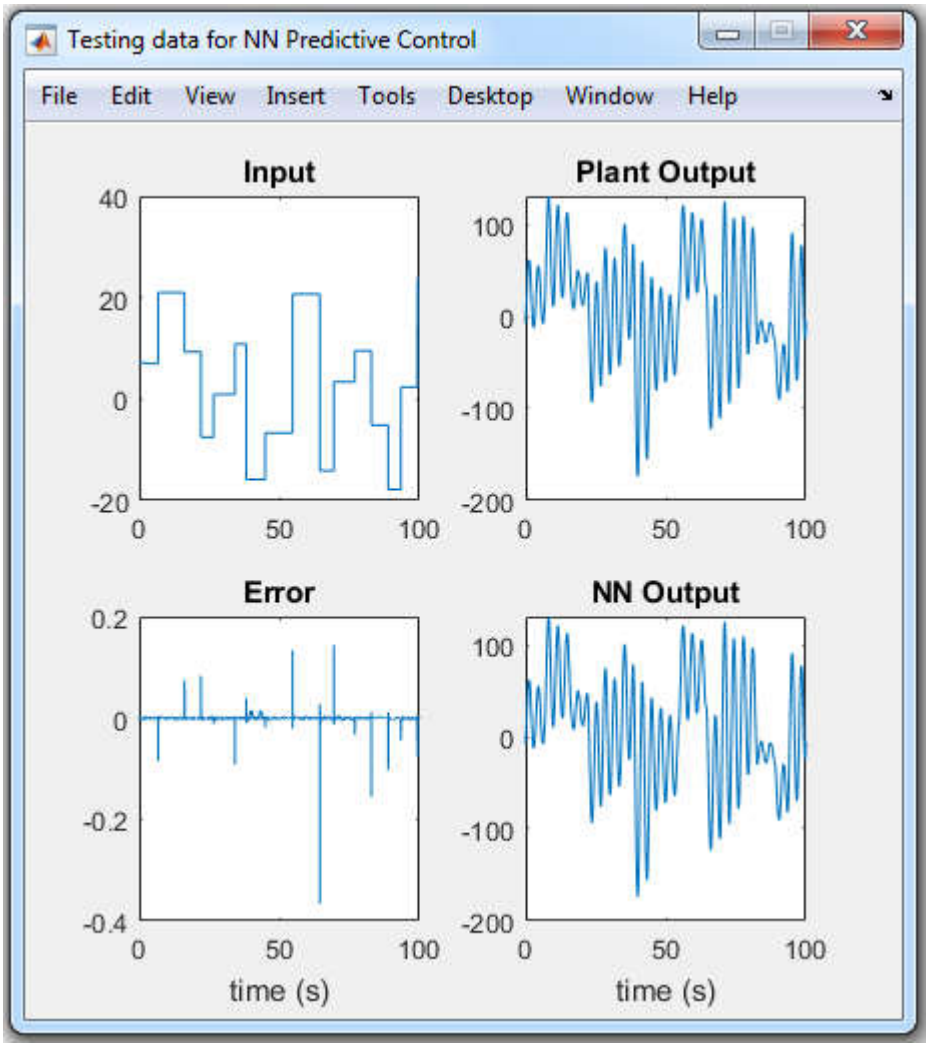


Рисунок 4.20 – Візуалізація результатів перевірки нейронної мережі на тестовій множині

Продовження опису вікна Plant Identification.

Після завершення процесу навчання у вікні **Plant Identification** з'являється повідомлення: *"Training complete. You can generate or import new data, continue training or save results by selecting OK or Apply"* (Навчання завершено. Можна згенерувати або імпортувати нові дані, продовжити навчання або зберегти результати, обравши кнопки **OK** або **Apply**).

На цьому етапі М-функція **Nnident** формує динамічну нейромережу **netn2** із заданою кількістю затримок на вході та виході моделі. При цьому попередньо набуті значення вагів і зміщень нейронів зберігаються. Елементи динамічної мережі показані на рис. 4.21.

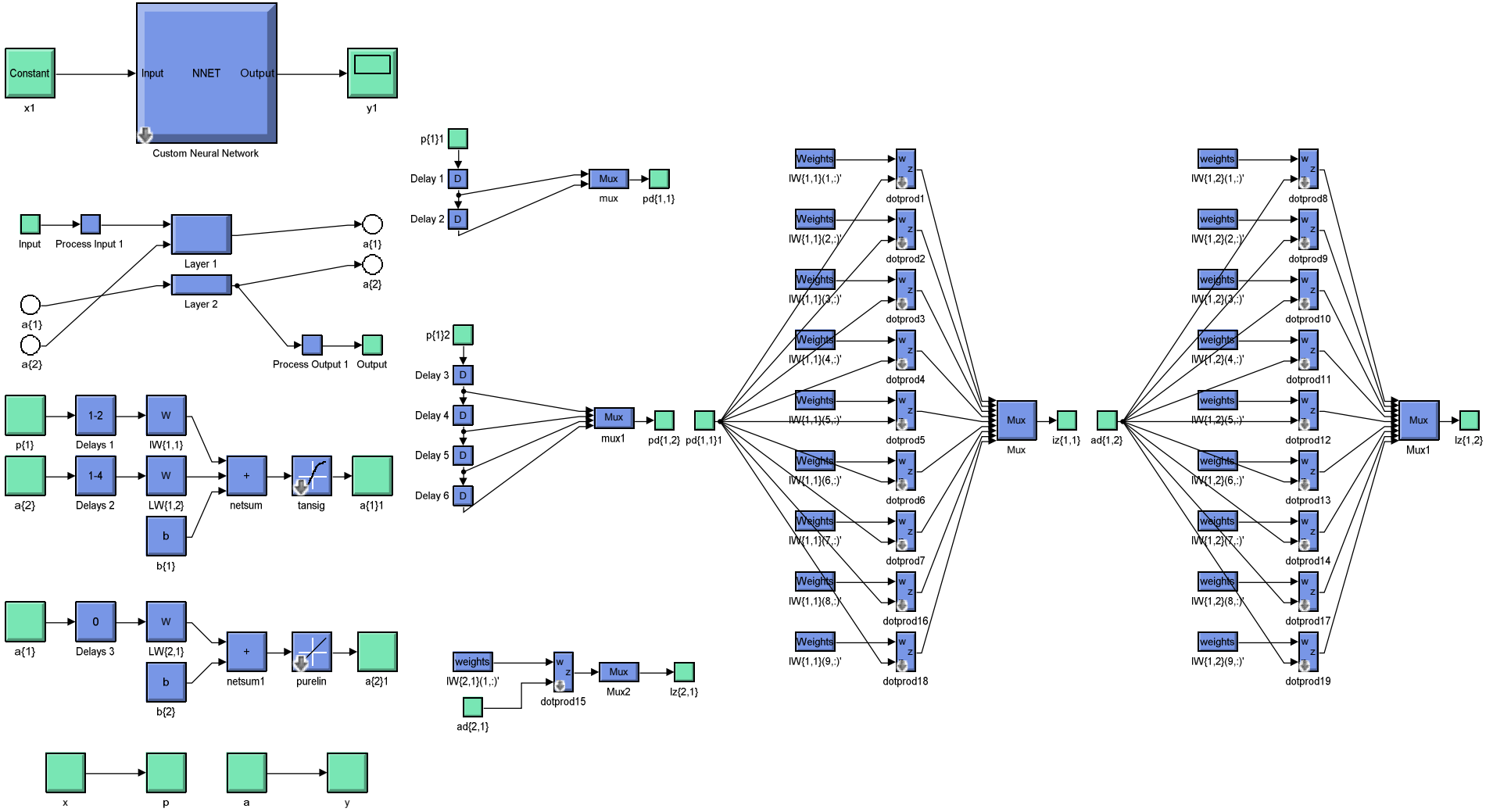


Рисунок 4.21 – Моделі елементів динамічної мережі з блоками затримок у Simulink

Модель динамічної мережі створюється у Simulink за допомогою команди **gensim(netn2)**. Кожен новий елемент стає доступним у окремому вікні після подвійного клацання на попередньому. На основі цих елементів формується повна схема динамічної мережі, показана на рис. 4.22.

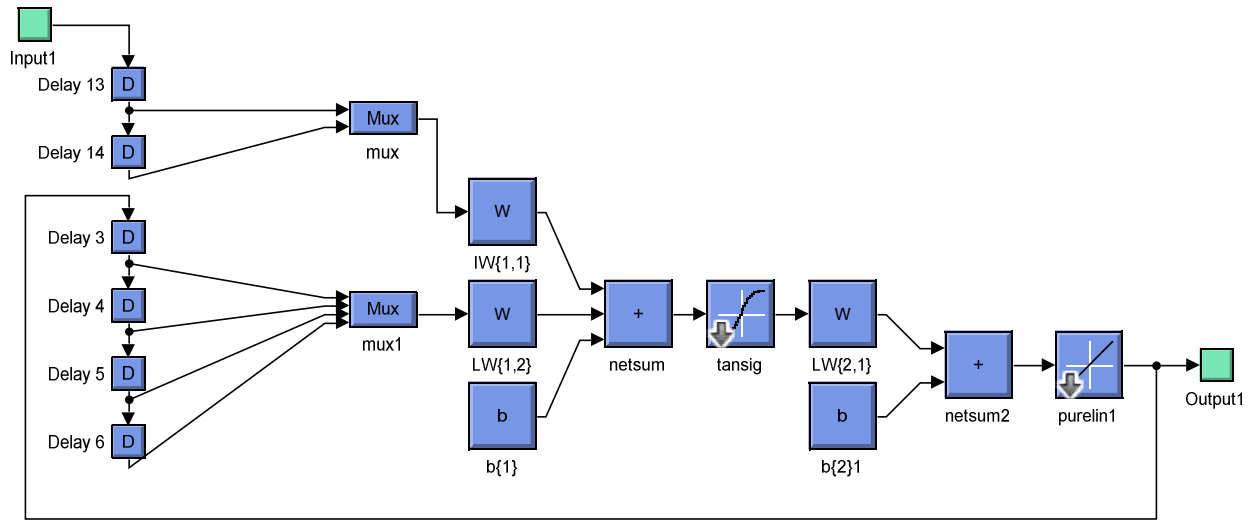


Рисунок 4.22 – Модель динамічної мережі з елементами затримок, побудованої в Simulink

Для успішної побудови моделей **netn** і **netn2** необхідно встановити **Breakpoint** у відповідному місці М-функції **Nnident.m**, оскільки після завершення виконання цієї функції подальше формування мереж стає неможливим.

Параметри нейромережевої моделі керованого об’єкта передаються до блоку **NN Predictive Controller** у Simulink. Мережа відображається як структурна схема, показана на рис. 4.23.

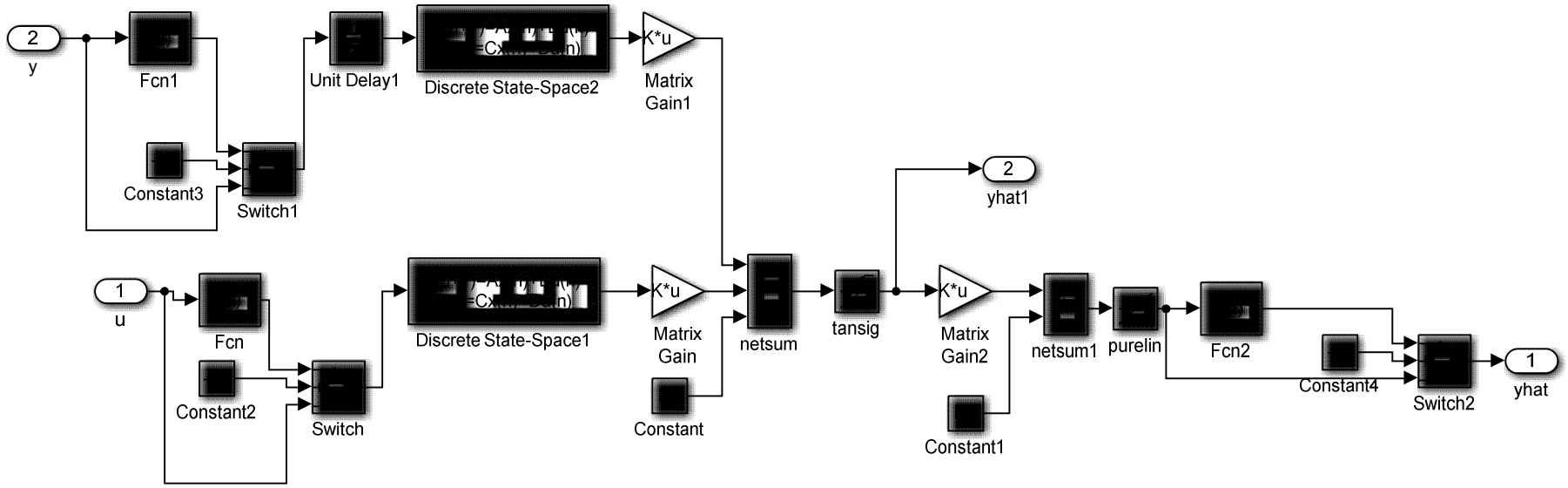


Рисунок 4.23 – Структурна схема нейромережевої моделі об'єкту регулювання

Блоки Matrix Gain і Matrix Gain1 відповідають матрицям IW{1,1} та LW{1,2}; блоки Constant (B1) і Constant1 (B2) – це зсуви нейронів першого та другого шарів; елементи затримок реалізовані блоками Discrete State Space 1 та Discrete State Space 2.

$$\mathbf{y}(n) = \mathbf{C}\mathbf{x}(n) + \mathbf{D}\mathbf{u}(n) \ ;$$

$$\mathbf{x}(n+1) = \mathbf{A}\mathbf{x}(n) + \mathbf{B}\mathbf{u}(n).$$

Візьмемо, для прикладу, $N_i = 2$ і $N_j = 5$. МВ цьому випадку атриці $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ і $\mathbf{D}$ мають наступний вид.

Discrete State Space 1

$$\mathbf{A} = \begin{vmatrix} 0 \end{vmatrix}, \quad \mathbf{B} = \begin{vmatrix} 1 \end{vmatrix}, \quad \mathbf{C} = \begin{vmatrix} 0 \\ 1 \end{vmatrix}, \quad \mathbf{D} = \begin{vmatrix} 1 \\ 0 \end{vmatrix}.$$

Discrete State Space 2

$$\mathbf{A} = \begin{vmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix}, \quad \mathbf{B} = \begin{vmatrix} 1 \\ 0 \\ 0 \\ 0 \end{vmatrix}, \quad \mathbf{C} = \begin{vmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}, \quad \mathbf{D} = \begin{vmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{vmatrix}.$$

Також у Simulink формується схема **ptest3sim2** (рис. 4.24), яка містить додаткові виходи та використовується М-функцією **predopt** для прогнозування процесу у майбутньому. При активації блоку **Subsystem** відкривається детальна схема, показана на рис. 4.25.

Об’єднана структурна схема нейрорегулятора, що поєднує моделі з рис. 4.6 і рис. 4.23, представлена на рис. 4.26.

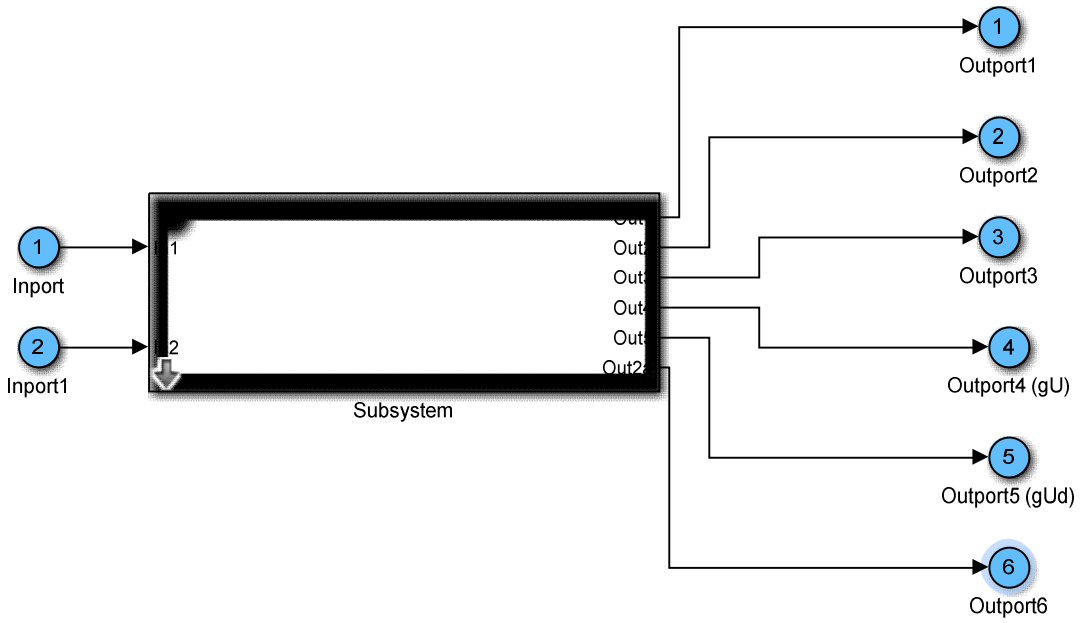


Рисунок 4.24 – Схема ptest3sim2, що використовується для прогнозування процесу функцією predopt

Після формування нейромережевої моделі керованого об’єкта користувач повертається до вікна **Neural Network Predictive Controller** (рис. 4.7), де здійснюється налаштування параметрів оптимізації.

Продовження опису вікна Neural Network Predictive Controller

Const Horizon (N2) – верхня межа підсумовування у функції якості (нижня межа фіксована і дорівнює 1). Цей параметр визначає горизонт прогнозу, тобто кількість кроків у майбутньому, на які алгоритм NNPC робить передбачення.

Збільшення Const Horizon (N2) дозволяє отримати більш далекі випереджальні прогнози, але водночас підвищує обчислювальну складність системи. Вибір оптимального значення цього параметра залежить від специфіки керованого об’єкта та вимог до точності прогнозу.

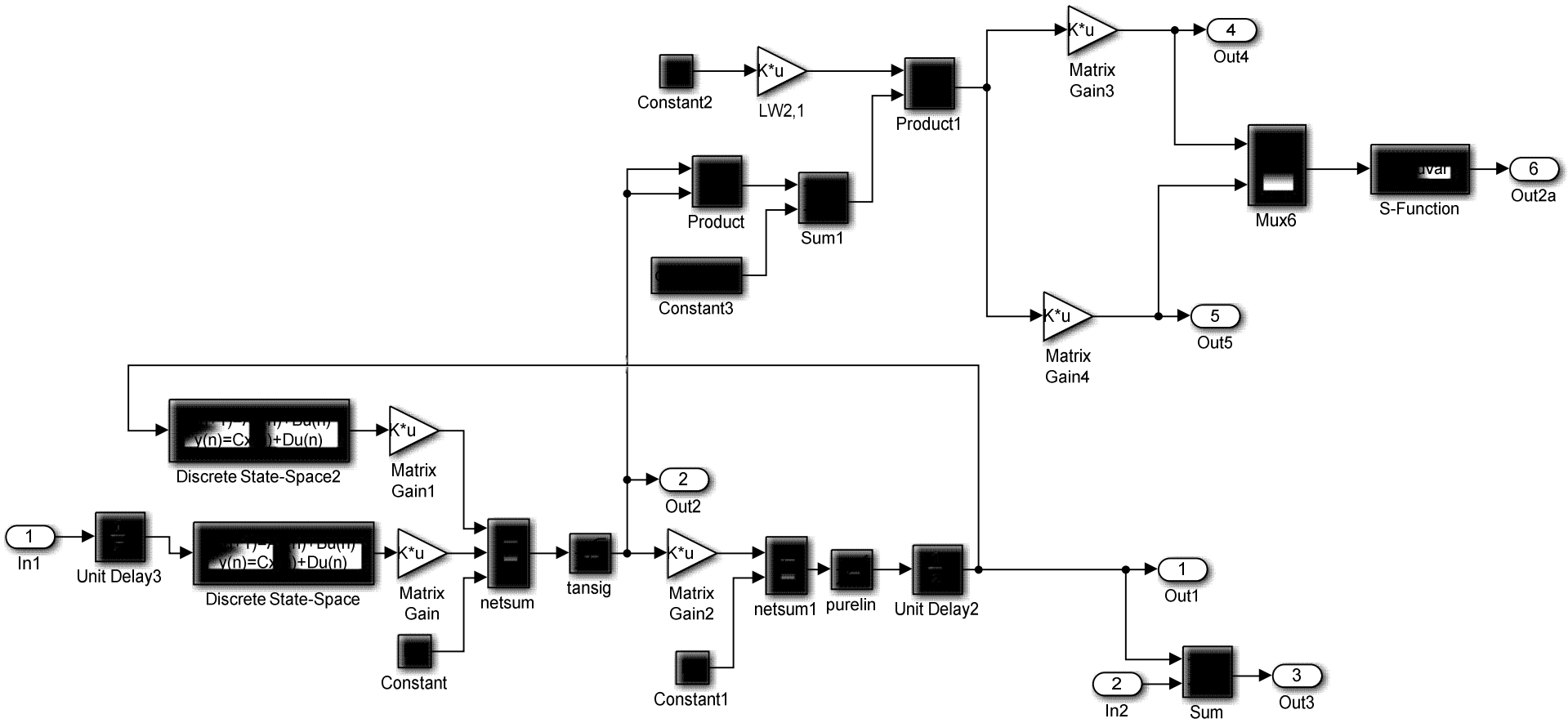


Рисунок 4.25 – Схема блоку Subsystem системи ptest3sim2

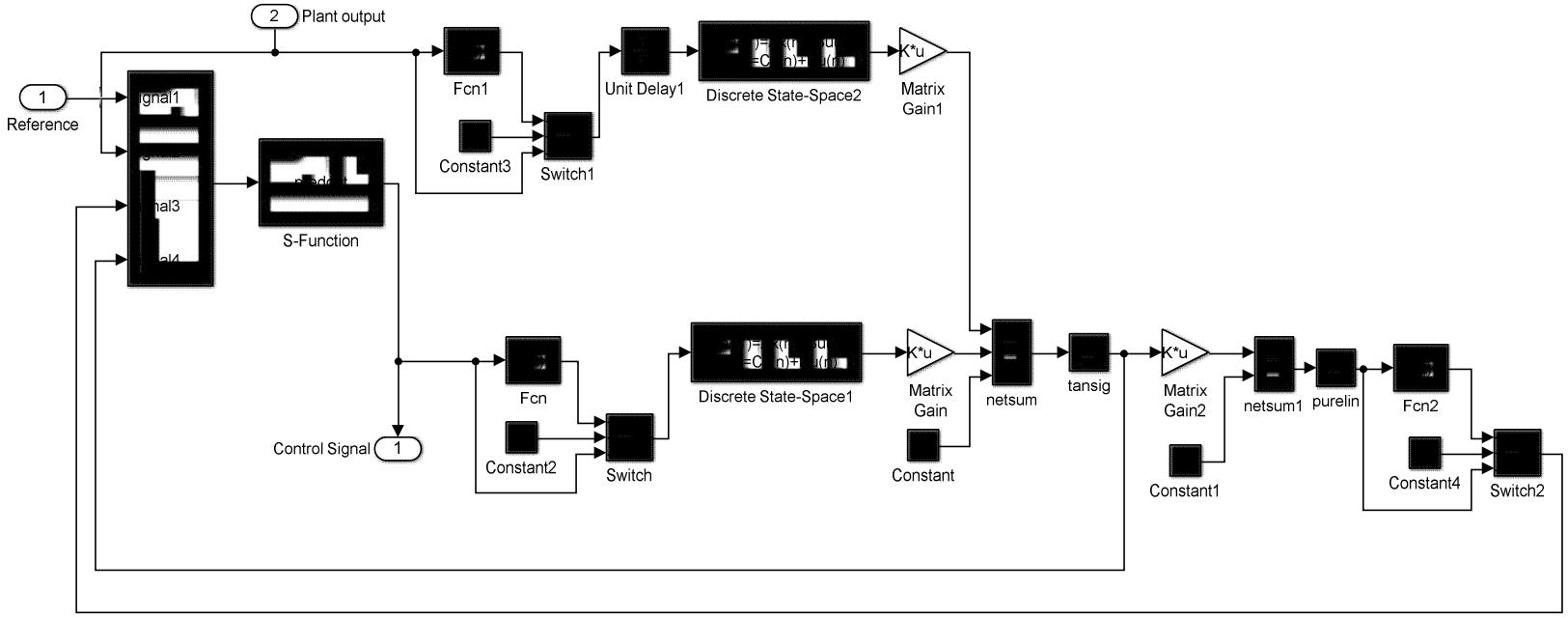


Рисунок 4.26 – Перетворена структурна схема нейрорегулятора

Control Horizon (Nu). Параметр **Control Horizon (Nu)** визначає верхню межу підсумовування при оцінці потужності керування. Він безпосередньо впливає на алгоритм прогнозного керування, визначаючи, на скільки кроків у майбутньому контролер буде розраховувати керуючі дії.

Основна ідея цього параметра полягає в тому, що збільшення Control Horizon дозволяє алгоритму оптимального керування враховувати більш віддалені майбутні стані системи, що сприяє точнішому досягненню цільових значень та підтриманню стабільності системи. Однак занадто велике значення Nu може призвести до підвищення обчислювального навантаження та збільшення вимог до ресурсів комп'ютера. Тому при виборі цього параметра слід враховувати характеристики системи, складність задачі та необхідний рівень прогнозу.

Control Weighting Factor ρ (Коефіцієнт ваги керування). Control Weighting Factor визначає вагу або важливість мінімізації керуючих сигналів у функції витрат контролера. Іншими словами, цей параметр дозволяє налаштувати, наскільки сильно контролер буде прагнути робити керуючі дії економічними або «дешевими» щодо витрат на управління.

Якщо коефіцієнт ρ встановлено великим, контролер надає пріоритет зменшенню амплітуди керуючих сигналів, що може бути корисно у випадках, коли управління дорого коштує або є фізичні обмеження. Зменшення значення коефіцієнта ρ призводить до того, що контролер більше концентрується на досягненні цільових значень системи, а економія керуючих сигналів стає менш пріоритетною.

Налаштування Control Weighting Factor дозволяє знайти оптимальний баланс між ефективністю керування та вартістю керуючих дій, що особливо важливо при застосуванні нейромережевого прогнозного контролю, де одночасно враховуються точність прогнозу і обмеження на ресурси управління.

Search Parameter α (Параметр пошуку). Search parameter – це параметр одновимірного пошуку, який задає поріг допустимого зменшення показника якості. Він використовується для контролю процесу оптимізації та визначає момент завершення пошуку, коли подальше покращення якості стає незначним.

Одновимірний пошук – це метод оптимізації, при якому оптимізується лише один параметр, тоді як інші залишаються фіксованими. У контексті нейромережевого прогнозного контролера цей метод допомагає налаштувати окремі параметри контрольної стратегії, що безпосередньо впливають на ефективність керування.

Поріг зменшення показника якості визначає мінімальне значення, яке повинен мати показник покращення, щоб вважати процес оптимізації завершеним. Коли приріст або зменшення показника якості стає меншим за цей поріг, алгоритм припиняє пошук, а отримані параметри вважаються оптимальними. Такий підхід дозволяє заощадити обчислювальні ресурси та прискорити збіжність оптимізації.

Minimization Routine (Процедура мінімізації). Minimization Routine визначає алгоритм оптимізації, який використовується для налаштування параметрів контрольної стратегії. Вибір конкретного методу оптимізації має значний вплив на точність і ефективність роботи контролера.

При виборі процедури враховуються особливості завдання керування, наявні обмеження, а також доступні обчислювальні ресурси. Ретельний підбір алгоритму дозволяє досягти оптимального балансу між точністю розрахунку керуючих сигналів і швидкістю обчислень.

Iterations Per Sample Time (Ітерації на крок часу). Параметр Iterations Per Sample Time визначає кількість ітерацій оптимізації, які виконуються протягом одного кроку часу системи (Sample Time). Він впливає на точність розрахунку оптимального керування та швидкість реакції контролера на зміни стану системи.

Збільшення кількості ітерацій дозволяє отримати більш точний розрахунок керуючих дій, проте підвищує обчислювальне навантаження. Навпаки, занадто мала кількість ітерацій може призвести до недостатньої точності оптимізації та втрати ефективності керування.

Оптимальне значення цього параметра зазвичай підбирається експериментально, виходячи з характеристик конкретної системи керування та обмежень

обчислювальних ресурсів. Воно забезпечує баланс між точністю оптимізації та швидкістю реакції контролера.

Після налаштування всіх параметрів оптимізації вони передаються в блок **NN Predictive Controller** у середовищі Simulink, де використовуються для розрахунку оптимальних керуючих сигналів під час роботи системи.

4.4 Вибір параметрів нейрорегулятора NN Predictive Controller

При проектуванні нейрорегулятора відбувається варіювання параметрів Const Horizon (N_2), Control Horizon (N_u) та Control Weighting Factor (ρ) (за умови, що нижня межа Const Horizon залишається зафіксованою і дорівнює 1), а також параметра одновимірного пошуку Search Parameter α, який визначає поріг допустимого зменшення показника якості, та кількість ітерацій оптимізації на один крок дискретності. Додатково обирається метод проведення одновимірного пошуку.

Дослідження показали, що параметри Const Horizon, Control Weighting Factor та обрана процедура пошуку мають незначний вплив на кінцеві результати синтезу регулятора. Тому для даної задачі були обрані значення: N_u=2, $\rho = 0,05$, $\alpha = 0,001$,а метод одновимірного пошуку – csrchbac.

Водночас значення Control Horizon (Nu) і кількість ітерацій на такт дискретності γ значно впливають на продуктивність контролера. Збільшення цих параметрів дозволяє підвищити точність керування, проте істотно зростає обчислювальне навантаження на кожному кроці. Для розв'язуваної задачі оптимальні значення знаходяться в межах $N_2 = 20 \div 30$, $\gamma = 2 \div 4$

При ідентифікації об'єкта керування критично важливим є вибір кількості нейронів у прихованому шарі N. Недостатня кількість нейронів унеможливлює виконання поставлених завдань, тоді як надмірна – призводить до перенавчання та збільшення часу обчислень. Для розглядуваної системи оптимальне значення

$N = 7 \div 12$, при цьому похибка на навчальній, контрольній та тестовій вибірках залишалася в межах $10^{-4} \div 10^{-6}$

Якість навчання мережі безпосередньо залежить від довжини навчальної вибірки $N_{\text{в}}$ та дискретного кроку часу Δt, який визначає інтервал між двома послідовними точками збору даних. Оптимальні значення для даної системи: $N_{\text{в}} = 20000$, $\Delta t = 0{,}05 \div 0{,}1\ c$. Збільшення Δt зменшує точність обчислень і скорочує різницю між помилкою на навчальному наборі та помилками на контрольній і тестовій множинах. Зменшення Δt потребує подовження навчальної вибірки $N_{\text{в}}$, що суттєво збільшує час навчання без значного зниження похибки.

Для формування репрезентативної вибірки важливо правильно встановити мінімальні та максимальні межі інтервалу ідентифікації, що залежать від динамічних характеристик об'єкта Subsystem. У даній роботі вони обрані як $t_{\min} = 4 \div 8\ c$, $t_{\max} = 10 \div 20\ c$.

При побудові нейромережевої моделі задається кількість затримок на вході z_1 та виході z_2 моделі. Найкращі результати були досягнуті при $z_1 = 2$, $z_2 = 2 \div 5$.

Результат навчання також залежить від початкових значень вагових коефіцієнтів w_{ij} і кількості навчальних циклів $N_{\text{ц}}$. Для досягнення глобального мінімуму навчання рекомендується багаторазово повторювати процес з різними початковими значеннями ваг w_{ij} та різними конфігураціями $N_{\text{ц}}$. У даній роботі для кожної конфігурації мережі використовувалася кілька сотень початкових точок, а кількість циклів, після якої похибка переставала зменшуватись, становила $300 \div 900$.

Для навчання мережі використано алгоритм Levenberg-Marquardt (trainlm), який забезпечує ефективну оптимізацію вагових коефіцієнтів і швидке зближення до мінімуму функції втрат.

4.5 Розрахунок динамічних характеристик систем з нейрорегулятором NN Predictive Controller

Побудова графіків перехідних процесів двомасової системи, керованої нейрорегулятором **NN Predictive Controller**,виконувалась з використанням моделей, створених в середовищі Simulink, що представлені на рис. 4.5 та 4.27.

На рис. 4.28–4.31 наведено графіки перехідних процесів основних змінних стану синтезованої системи. Аналіз отриманих результатів показує, що система демонструє задовільну реакцію на ступінчасті впливи з випадковими амплітудами. Перехідні процеси мають коливальний характер із невеликим перерегулюванням, що свідчить про стабільність та адекватність регулятора.

Таким чином, синтезований нейромережевий регулятор з прогнозом NN Predictive Controller може бути ефективно використаний для керування двомасовою електромеханічною системою, розглянутою в даній роботі.

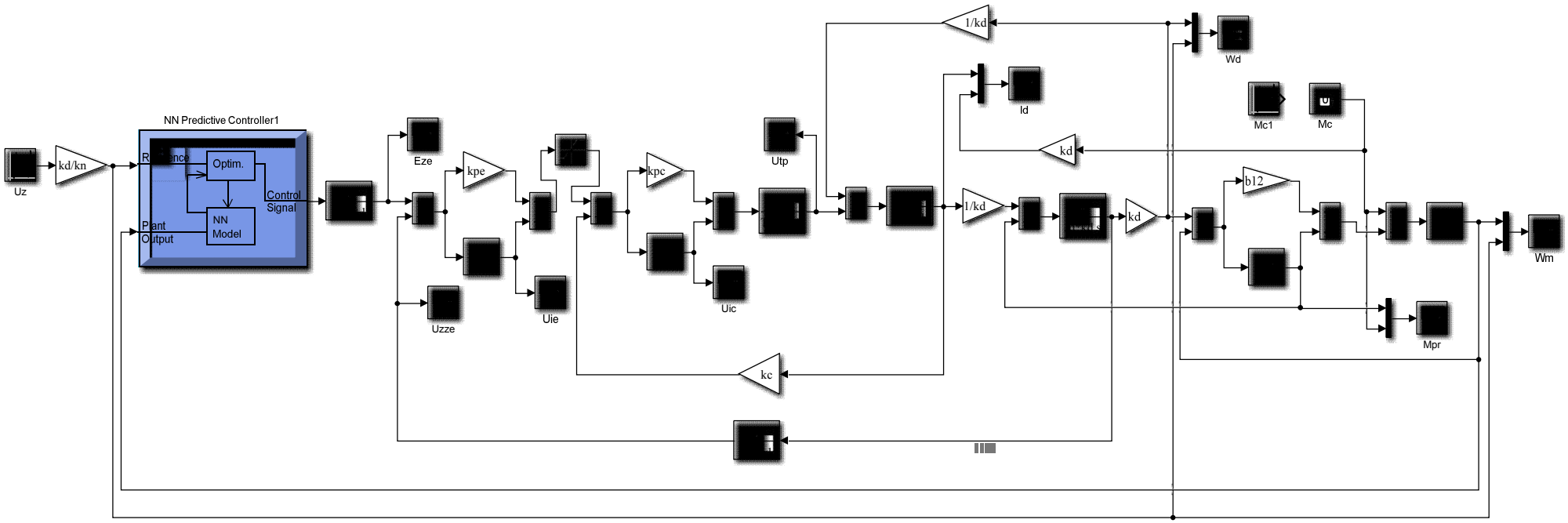
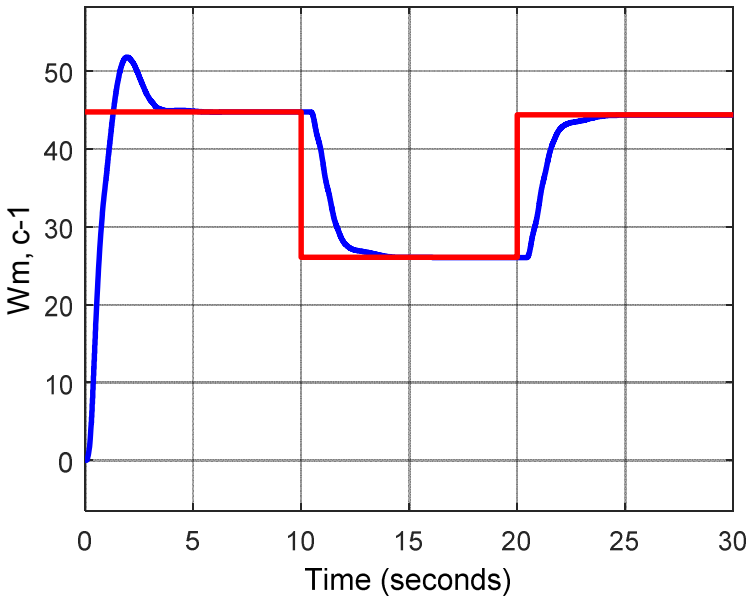
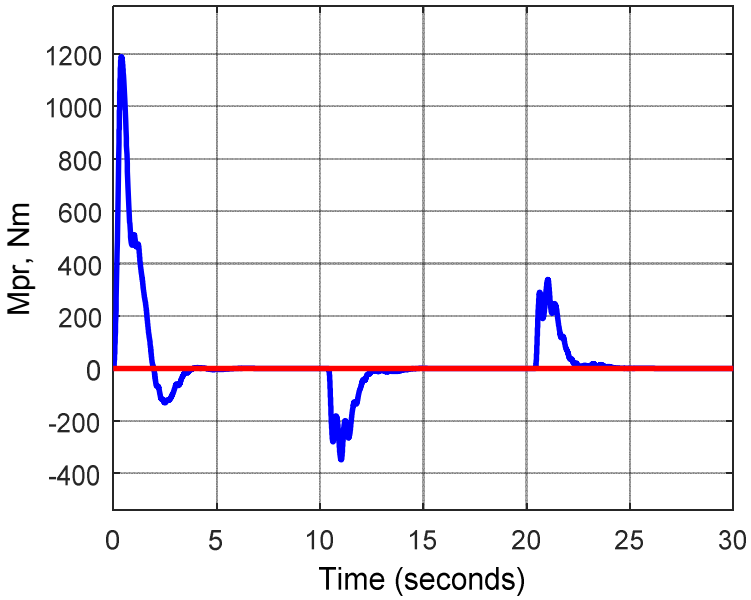


Рисунок 4.27 – Блок-схема моделі двомасової системи із застосуванням нейрорегулятора



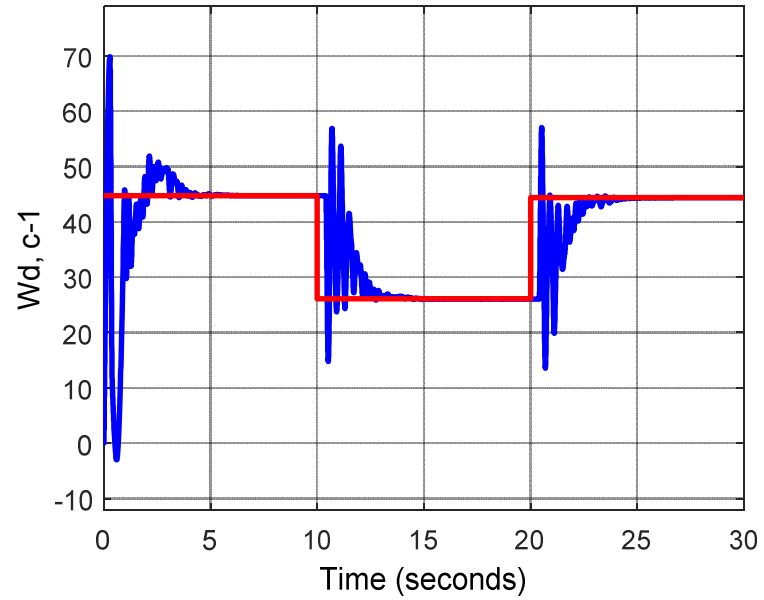
а)



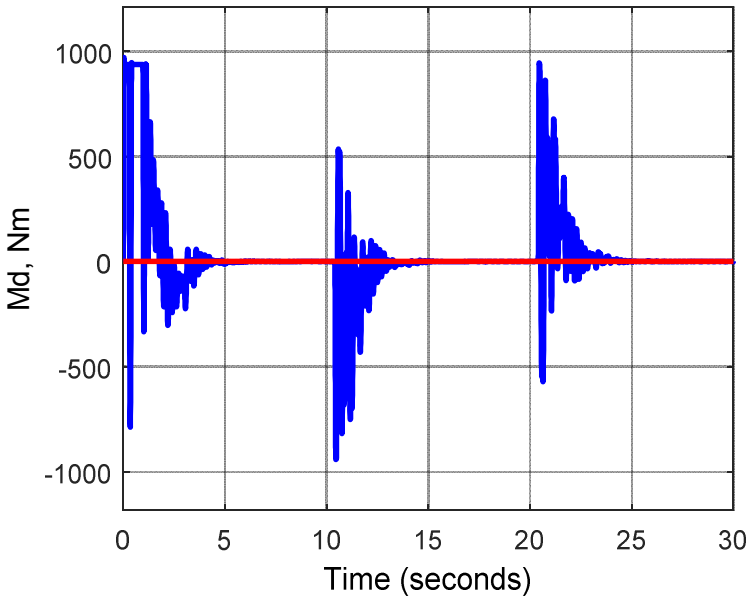
б)

Рисунок 4.28 – Перехідні процеси механічних параметрів двомасової системи з нейрорегулятором під дією задаючої величини:

а) швидкість механізму $\omega_{м} = f(t)$, б) момент пружності $M_{пр} = f(t)$



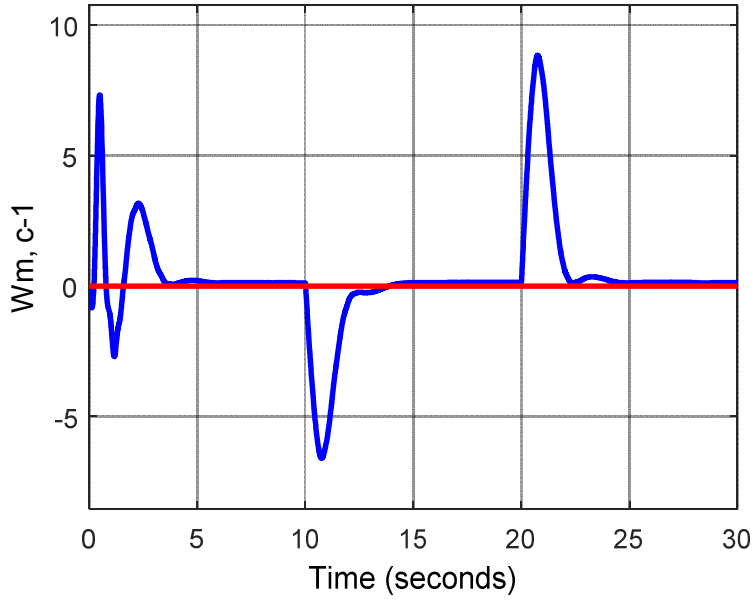
а)



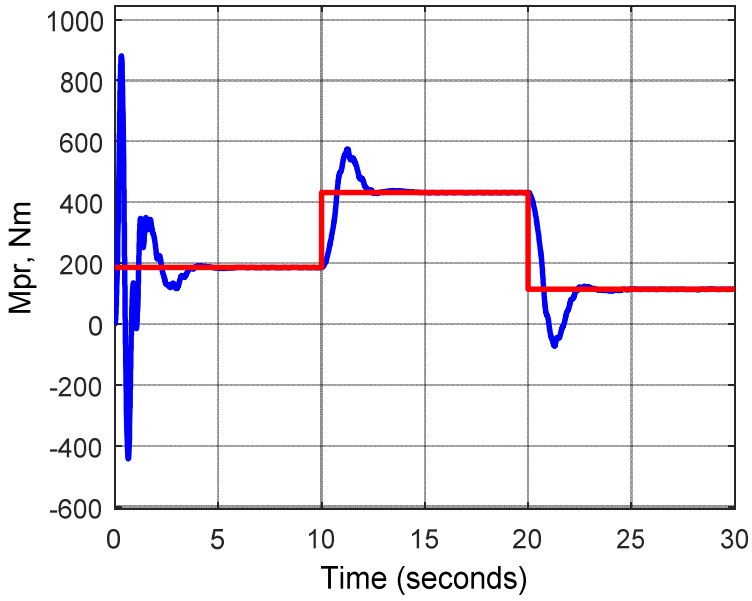
б)

Рисунок 4.29 – Динаміка двигуна двомасової системи з нейрорегулятором під впливом задаючої величини:

а) швидкість обертання двигуна $\omega_д = f(t)$, б) момент двигуна $M_д = f(t)$



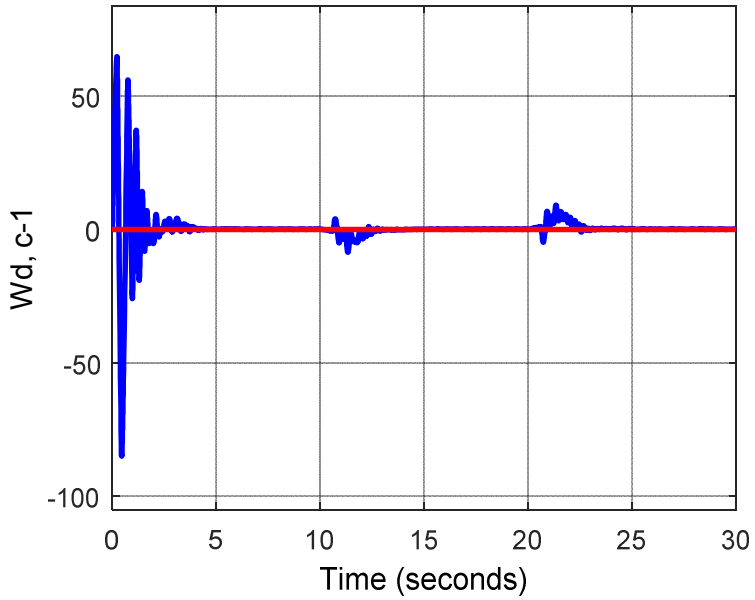
а)



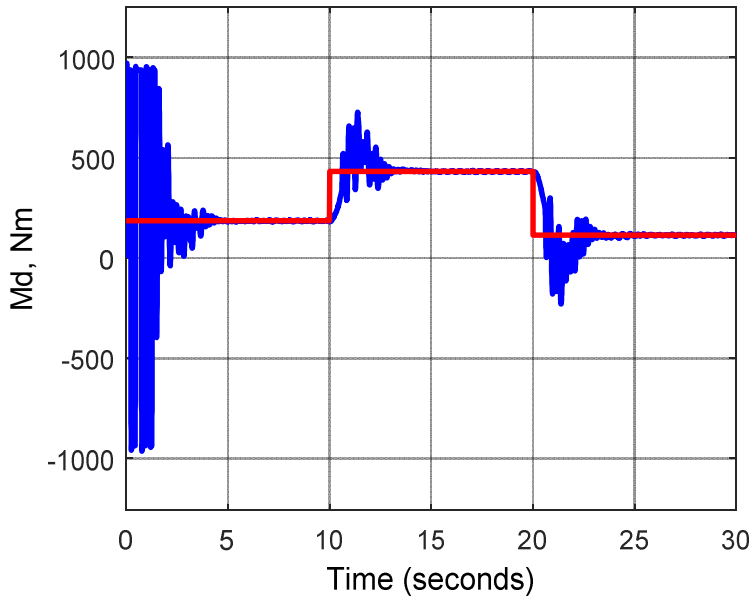
б)

Рисунок 4.30 – Перехідні процеси механічних параметрів двомасової системи з нейрорегулятором під дією зовнішнього збурення:

а) швидкість механізму $\omega_{\text{м}} = f(t)$, б) момент пружності $M_{\text{пр}} = f(t)$



а)



б)

Рисунок 4.31 – Перехідні процеси механічних параметрів двомасової системи з нейрорегулятором під дією зовнішнього збурення:

а) швидкість механізму $\omega_{м} = f(t)$, б) момент пружності $M_{пр} = f(t)$

ВИСНОВКИ ДО РОЗДІЛУ 4

1. Виконано детальний аналіз принципів побудови та роботи нейрорегуляторів, реалізованих у пакеті прикладних програм Neural Network Toolbox системи MATLAB. Розглянуто три основні типи регуляторів: регулятор з прогнозуванням (NN Predictive Controller), регулятор на основі моделі авторегресії з ковзним середнім (NARMA-L2 Controller) та регулятор, побудований на базі еталонної моделі (Model Reference Controller). Для задачі управління механізмом переміщення моста мостового крана у цій роботі обрано нейрорегулятор з прогнозуванням NN Predictive Controller. Визначено його структуру та поетапну методику синтезу, що включає налаштування параметрів навчання та обробку сигналів з об’єкта управління.

2. У середовищі SIMULINK системи MATLAB побудована модель системи управління, в якій використано нейрорегулятор NN Predictive Controller. Модель включає два основні блоки: блок керованого об’єкта, представленого схемою моделі двомасової системи управління механізмом переміщення моста, та блок регулятора NN Predictive Controller. Така структура дозволяє імітувати реальні умови роботи електроприводу та оцінити якість регулювання в різних режимах експлуатації.

3. Проведено процес генерації навчальної послідовності шляхом варіювання параметрів моделі виконавчого пристрою. Це дозволило забезпечити високоточне навчання нейромережі. Оптимальні параметри навчальної послідовності встановлені наступним чином: довжина навчальної вибірки – 10 000 елементів; інтервал між послідовними зніманнями даних – 0,2 с; мінімальний інтервал для ідентифікації – 20 с; максимальний інтервал для ідентифікації – 50 с. Завдяки таким налаштуванням нейромережа отримала достатню інформацію для точного відтворення динаміки об’єкта управління.

4. Створено двошарову нейронну мережу з прямим проходженням сигналу (feedforward), яка пройшла навчання із застосуванням алгоритму Левенберга–Марковдта. Було визначено оптимальну кількість нейронів у прихованому

шарі для досягнення мінімальної помилки навчання. Результати показали, що середня помилка навчання має дуже мале значення, миттєві помилки на навчальній, контрольній та тестовій множинах не перевищують 1, а коефіцієнт кореляції між виходом мережі та реальними даними R дорівнює 1. Це свідчить про високий ступінь точності відтворення моделі об’єкта.

5. Виконано синтез нейрорегулятора з прогнозуванням NN Predictive Controller. Під час варіювання параметрів регулятора встановлено, які з них мають найбільший вплив на якість регулювання, а також визначено оптимальні значення для забезпечення стабільної роботи системи. Використання нейромережевої моделі об’єкта з високим ступенем ідентифікації дозволило побудувати регулятор, що забезпечує високу точність слідування за задаючим сигналом та ефективне гасіння коливань у системі.

6. Проведено моделювання роботи системи з синтезованим нейрорегулятором у середовищі MATLAB/SIMULINK. Аналіз перехідних процесів показав, що в умовах пуску та при різких змінах навантаження система демонструє задовільний динамічний характер. Нейрорегулятор забезпечує астатичну роботу системи: перерегулювання не перевищує 10%, час виходу на сталий режим – 4 с, що повністю відповідає заданим вимогам до роботи електроприводу мостового крана.

ВИСНОВКИ

У магістерській роботі розв’язано актуальну науково-практичну задачу, що полягає у розробці нейромережевої системи управління багатомасовою електромеханічною системою механізму переміщення моста мостового крана. Розроблена система забезпечує високу точність регулювання та ефективну динаміку роботи електроприводу.

Основні результати роботи можна сформулювати таким чином:

1. Проведено аналіз сучасних систем управління електроприводами основних механізмів мостових кранів з точки зору забезпечення високої якості регулювання. Виявлено, що застосування нейромережевих технологій є перспективним напрямком для управління багатомасовими електромеханічними системами, оскільки вони дозволяють враховувати складні нелінійності і динаміку об’єкта, що значно підвищує ефективність роботи системи.

2. На основі технічних характеристик мостових кранів та вимог до електроприводів основних механізмів обрано електропривод постійного струму, виконаний за системою ТП-Д. Виконано вибір електродвигунів і основного силового обладнання електроприводу, а також перевірку їх роботи щодо нагріву та перевантажувальної здатності, що дозволяє забезпечити безпечну та надійну експлуатацію.

3. Розроблено структуру системи автоматичного управління електроприводом із застосуванням двоконтурної системи підлеглого регулювання: внутрішній контур – струму, зовнішній – ЕРС. Виконано розрахунок параметрів силового ланцюга, необхідних для побудови системи автоматичного регулювання. Проведено синтез послідовних коректуючих пристроїв: контур струму оптимізований за модульним критерієм, а контур ЕРС – за симетричним. Здійснено моделювання роботи системи на ЕОМ із використанням пакету MATLAB, що підтвердило правильність обраної структури та параметрів.

4. Розроблено математичну модель двомасової механічної системи, яка враховує пружність реальних механічних зв’язків між двигунами та мостом.

Моделювання показало, що при переходових процесах основні координати системи демонструють виражені коливання, що обумовлено пружною поведінкою механічної частини.

5. Для забезпечення бажаних динамічних характеристик розроблено систему управління з нейрорегулятором NN Predictive Controller, реалізованим у пакеті Neural Network Toolbox системи MATLAB. Викладено принцип побудови нейрорегулятора, його алгоритмічну структуру та порядок синтезу, а також проведено моделювання процесів у двомасовій системі. Аналіз результатів показав, що реакція системи на ступінчасту задачу з випадковою амплітудою є задовільною, із швидким загасанням коливань. Система демонструє астатичну поведінку: перерегулювання не перевищує 10%, а час виходу на сталий режим – 4 секунди, що відповідає заданим технічним вимогам.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Штучні нейронні мережі: навчальний посібник / С. В. Ткаліченко. – Кривий Ріг: Державний університет економіки і технологій , 2023. –150 с.
2. Технології нейронних мереж і нечіткого моделювання в системах управління: підруч. для здобувачів вищої освіти спец. 151 Автоматизація та комп'ютерно-інтегровані технології / Г.І. Канюк, Б.І. Кузнецов, Т.Ю. Василець, А.Ю. Мезеря, О.О. Варфоломієв. – Харків : Друкарня Мадрид, 2020. – 306 с.
3. Нейромережеві технології в системах управління: Підручник для візів./ Б. І. Кузнецов, Т.Ю. Василець, Т.Б. Нікітіна, В. В. Коломиєць, О.О. Варфоломієв; Укр. інж.-пед. акад.. - Харків: УІПА, 2014. - 232 с.
4. Кирик В. В. Математичний апарат штучного інтелекту в електроенергетичних системах:підручник..– Київ: КПІ ім. Ігоря Сікорського, Вид-во «Політехніка», 2019. –224 с.
5. Зайченко Ю.П. Основи проектування інтелектуальних систем.. – К.: Слово, 2004. –353 с.
6. Штучні нейронні мережі: навчальний посібник для студентів вищих навчальних закладів / О.Г.Руденко, Є.В. Бодянський. – К: Компанія СМІТ, 2006, 404 с.
7. Субботін С. О. Нейронні мережі : теорія та практика: навч. посіб. / С. О. Субботін. – Житомир : Вид. О. О. Євенок, 2020. – 184 с.
8. Тимощук П.В. Штучні нейронні мережі; Навч. посібн. - Львів: Львівська політехніка, 2011. -444 с.
9. Кононюк, А. Ю. .Нейронні мережі і генетичні алгоритми. - Київ: Корнійчук, 2008. - 468 с.
10. Кирик В. В. Математичний апарат штучного інтелекту в електроенергетичних системах:підручник..– Київ: КПІ ім. Ігоря Сікорського, Вид-во «Політехніка», 2019. –224 с.
11. Зайченко Ю.П. Основи проектування інтелектуальних систем.. – К.: Слово, 2004. –353 с.

12. Штучні нейронні мережі: базові положення Навчальний посібник Електронне мережне навчальне видання / І.А. Терейковський, Д.А. Бушуєв, Л.О. Терейковська, Київ: КПІ ім. Ігоря Сікорського, 2022.- 123 с.

13. McCulloch W. S., Pitts W. A logical calculus of ideas imminent in nervous activity // Bulletin Mathematical Biophysics. – 1943. – № 5. – P.115-133 (Имеется перевод: Дж. Маккаллок, У. Питтс. Логическое исчисление идей, относящихся к нервной деятельности. В кн.: Автоматы. – М.: ИЛ, 1956).

14. Rosenblatt F. The perceptron: A probabilistic model for information storage and organization in the brain // Psychological Review.

15. Рябинин А. Д., Шквар А. М. Некоторые принципы функционального построения инвариантных бионических систем управления // Тр. 4-го Всесоюз. совещания. "Теория инвариантности и теория чувствительности автоматических систем". Ч. II. 26 - 30 апреля 1971.- Киев, 1971.- С 54-64.

16. Шквар А. М. Функциональные и структурные аспекты построения некоторых нейронных структур управления // Тр. 4-го Всесоюз. совещания "Теория инвариантности и теория чувствительности автоматических систем". Ч. II. 26-30 апреля 1971. – Киев, 1971. – С. 78-89.

17. Werbos P. J. Beyond regression: New tools for prediction and analysis in the behavioral sciences. PhD Thesis, Harvard University, Cambridge, MA. – 1974.– P. 124-137.

18. Rumelhart D. E., Hinton G E., Williams R. J. Learning internal representation by error propagation // In: D.E. Rumelhart, J.L. McClelland (Eds.) Parallel Distributed Processing, Vol. I Foundations. – Cambridge, MA: MIT Press, 1986. – P. 318-362.

19. Rumelhart D. E., Hinton G. E., Williams R. J. Learning representation by back-propagating errors // Nature. – 1986. – Vol.323. – P. 533-536.

20. Narendra K.S., Parthasarathy K. Identification and control of dynamical system using neural networks // IEEE Trans. Neural Networks. – 1990. – Vol.1. – №1. – P. 4-27.

21. Narendra K.S., Parthasarathy K. Gradient methods for the optimization of dynamical systems containing neural networks // IEEE Trans. Neural Networks. – 1991. – Vol.2. – № 2. – P.252-262.

22. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). *Learning representations by back-propagating errors. Nature*, 323(6088), 533–536.

23. Kohonen, T. (1990). *The self-organizing map. Proceedings of the IEEE*, 78(9), 1464–1480

24. Hopfield, J. J. (1991). *Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences*, 79(8), 2554–2558

25. LeCun, Y., Bengio, Y., & Hinton, G. (2015). *Deep learning. Nature*, 521(7553), 436–444.

26. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

27. Silver, D., Schrittwieser, J., Simonyan, K., et al. (2017). *Mastering the game of Go without human knowledge. Nature*, 550, 354–359.

28. Li Y., Sundararajan N., Saratchandran P. (2001), Neuro-controller design for nonlinear fighter aircraft maneuver using fully tuned RBF networks, Automatica, Vol. 37, No 8, pp. 1293-1301.

29. Gundy-Burlet K., Krishnakumar K., Limes G., Bryant D. (2004), Augmentation of an Intelligent Flight Control System for a Simulated C-17 Aircraft, J. of Aerospace Computing, Information, and Communication, Vol. 1, No.12, pp. 526-542.

30. Nikiforova L. N., Petrosyan E. A., Yakemenko G. V. (2000), "Neyrokompyuteryi v upravlenii vertoletami", [Neurocomputers in helicopter control], Artificial Intelligence, No. 3, pp. 290-298.

31. Kupin A. I. (2008), "Intelektualna IdentifIkatsIya ta keruvannya v umovah protsesiv zbagachuvalnoyi tehnologiyi", [Intelligent identification and control in the processes of enrichment technology], KTU, Krivoy Rog, 204 p.

32. D. Gu and H. Hu. (2002), Neural Predictive Control for a Car-like Mobile Robot, International Journal of Robotics and Autonomous Systems, Vol. 39 (2), pp. 73-86.

33. Danil V. Prokhorov. (2008), Toyota Prius HEV Neurocontrol and Diagnostics, Neural Networks, No. 21, pp. 458-465.
34. Zmeu K. V., Markov N. A., Shipitko I. A., Notkin B. S. (2009), “Bezmodelnoe prognoziruyuschee inversnoe neyro-upravlenie s regeneriruemyim etalonnyim perehodnyim protsessom”, [A modelless predictive inverse neuropravlenie with a regenerated reference transient process], Intelligent Systems, No. 3, pp. 109-117.
35. Kuznetsov B. I. , Vasilets T. Yu., Varfolomiev O. O. (2015), “Neyromerezheva sistema navedennya i stabIlIzatsiyi z regulyatorom na osnovi etalonnoyi modeli”, [Neural network aiming and stabilization system with the reference model controller], Electrical engineering and electromechanics, No. 4, pp.35-39.
36. Dzyuba D. A., Chernodub A. N. (2010), “Primenenie metoda kontroliruemogo vozmuscheniya dlya modifikatsii neyrokontrollerov v realnom vremeni”, [The application of the controlled disturbance method for the modification of real time neural controllers], Mathematical machines and systems, No. 4, pp. 20-28.
37. Dias F.M., Mota A.M. (2001), Comparison between Different Control Strategies using Neural Networks, 9th Mediterranean Conference on Control and Automation, Croatia, Dubrovnik.
38. Venayagamoorthy G.K., Harley R.G., Wunsch D.C. (2003), Implementation of Adaptive Criticbased Neurocontrollers for Turbogenerators in a Multimachine Power System”, IEEE Transactions on Neural Networks, Vol. 14, Issue 5, pp. 1047–1064.
39. D’Emilia G., Marrab A., Natalea E. (2007), Use of neural networks for quick and accurate autotuning of PID controller, Robotics and Computer-Integrated Manufacturing, Vol. 23, pp. 170 – 179.
40. Ferreira,P.M., Ruano,A.E., Silva,S., Conceição,E.Z.E. (2012). Прогнозоване керування на основі нейронних мереж для теплового комфорту та енергозбереження у громадських будівлях. Energy & Buildings,55, 238-251.
41. Yadav,M.K., Chandra,D. (2015). Керування системами кондиціонування повітря за допомогою нейронної мережі. Міжнародний журнал електроніки та електротехніки, 3(5), 406-410.
42. Камінскі, М. та ін. (2023). Застосування нейронних мереж в електроприводах - тенденції та перспективи. Energies, 16 (11).

43. Jin,L., Li,S., Yu,J., He,J. (2018). Керування роботом-маніпулятором за допомогою нейронних мереж: Огляд. Neurocomputing,285, 23-34.

44. Лі, Т., Чжан, Г., Чжан, Т., Пан, Дж. (2024). Адаптивне нейромережеве відстеження керування роботизованими маніпуляторами на основі спостерігача збурень. Processes, 12(3), 499.

45. Liu,Z., Peng,K., Han,L., Guan,S.C. (2023). Моделювання та керування роботизованими маніпуляторами на основі штучних нейронних мереж: Огляд. Іранський журнал науки і техніки, Transactions of Mechanical Engineering, 47, 1307-1347.

46. Лі, Дж., Су, Дж., Ю, В., Мао, Х., Лю, З., Фу, Х. (2024).Рекурентна нейронна мережа для керування траєкторією маніпулятора з невідомою матрицею мас. Frontiers in Neurorobotics.

47. Гупта, П., Сінха, Н. К. (2017). Керування роботизованими маніпуляторами за допомогою нейронних мереж: Огляд. У Методах та застосуваннях інтелектуального керування, 103-136. Springer.

48. Sun S., Cao Z., Zhu H., Zhao J. A survey of optimization methods from a machine learning perspective // *IEEE Transactions on Cybernetics*. – 2020. – Vol. 50, No. 8. – P. 3668–3681.

49. Zhao R., Yan R., Chen Z., Mao K., Wang P., Gao R. X. Deep learning and its applications to machine health monitoring // *Mechanical Systems and Signal Processing*. – 2019. – Vol. 115. – P. 213–237.

50. Rawat W., Wang Z. Deep convolutional neural networks for image classification: A comprehensive review // *Neural Computation*. – 2017. – Vol. 29, No. 9. – P. 2352–2449.

51. Lim B., Zohren S. Time-series forecasting with deep learning: A survey // *Philosophical Transactions of the Royal Society A*. – 2021. – Vol. 379, No. 2194. – Article 20200209.

52. Liu H., Lang B., Liu M., Yan H. Machine learning and deep learning methods for cybersecurity // *IEEE Access*. – 2020. – Vol. 8. – P. 139 101–139 126.

53. Liu, H., Lang, B., Liu, M., Yan, H. Machine Learning and Deep Learning

Methods for Intrusion Detection Systems: A Survey // Applied Sciences. – 2019. – Vol. 9, No. 20. – P. 4396. – MDPI

54. Aljawarneh, S., Aldwairi, M., Yassein, M. B. A Survey of Neural Networks Usage for Intrusion Detection Systems // Journal of Ambient Intelligence and Humanized Computing. – 2021. – Vol. 12, No. 3. – P. 3153–3172.

55. Kober J., Bagnell J. A., Peters J. Reinforcement learning in robotics: A survey // *The International Journal of Robotics Research.* – 2013. – Vol. 32, No. 11. – P. 1238–1274.

56. Esteva A., Robicquet A., Ramsundar B., et al. A guide to deep learning in healthcare // *Nature Medicine.* – 2019. – Vol. 25. – P. 24–29.

57. Miotto, R., Wang, F., Wang, S., Jiang, X., Dudley, J. T. A Review of Deep Learning Algorithms and Their Applications in Healthcare // Journal of Biomedical Informatics. – 2018. – Vol. 83. – P. 168–185.

58. Young, T., Hazarika, D., Poria, S., Cambria, E. Recent trends in deep learning based natural language processing // IEEE Computational Intelligence Magazine. – 2018. – Vol. 13, No. 3. – P. 55–75.

59. Runge, J., et al. Renewable energy forecasting: A review // Renewable and Sustainable Energy Reviews. – 2019. – Vol. 120. – 109642.

60. Kuster, C., Rezgui, Y., Mourshed, M. A Scoping Review of Deep Neural Networks for Electric Load Forecasting // Energy Informatics. – 2021. – Vol. 4, No. 1. – P. 1–26.

61. Amasyali, K., El-Gohary, N. M. A Review of Deep Learning Techniques for Forecasting Energy Use in Buildings // Energy and Buildings. – 2018. – Vol. 177. – P. 535–548.

62. Jain, A., Kumar, S. Deep learning in manufacturing: A review of applications, challenges, and future directions // Journal of Manufacturing Systems. – 2022. – Vol. 63. – P. 224–249.

63. Li, X., Zhang, W., Chen, X. Deep Learning Techniques in Intelligent Fault Diagnosis and Prognosis for Industrial Systems: A Review // Sensors. – 2023. – Vol. 23, No. 3. – P. 1305. – MDPI.

64. Wang, Y., Chen, Z., Li, H., et al. Data-Driven Machinery Fault Detection: A Comprehensive Review // Mechanical Systems and Signal Processing. – 2024. – Vol. 210. – 110013.

65. Cao, Y., Gu, Q. A survey on generative AI for art, design, and media // ACM Computing Surveys. – 2023. – Vol. 55, No. 12. – P. 1–35

66. Narendra Narendra K. Gallman P.O. An iterative method for the identification of nonlinear system using a Hammerstein model // IEEE Trans. Aut. Control. - 1966. - Vol. 11. -N3. - P.546-550.

67. Psaltis, D., Sideris, A., Yamamura, A. A Multilayered neural network controller // IEEE Control Systems Magazine. - 1988. - Vol.8. - N4. -P. 17-21.

68. Joardan M.A. Generic constraints on underspecified target trajectories // Proc. Of Int. Joint Conf. On Neural Networks (IJCNN), Washington. - 1989.- Vol.1. -N3. - P.217-225.

69. Joardan M.A., Rumelhart D.E. Forward models: Supervised learning with a distal teacher // Cognitive Science.- 1990. -Vol.16. -N5.-P.313-355.

70. Saerens M, Soquet A. Neural controller based on backpropagation algorithm // IEEE Proceedings-F. - 1991. -Vol.138. -N.I. -P.55-62.

71. Miller III, W.T., Glanz, F.H., Kraft III, L.G. Application of a general learning algorithm to control of robotic manipulators // Int. J. of Robotics Research. - 1987. - Vol.6. - N2. - P.84-98.

72. Kawato, M., Furukawa, K, Suzuki, R. A hierarchical neural network model for control and learning of voluntary movement//Biological Cybernetics. - 1987. - Vol.57. - N2. -P. 169-185.

73. Pham D.T., Liu X. Neural Network for Identification, Prediction and Control. - London: Springer - Verlag, 1997. - 238p.

74. Milito R., Padilla C.S., Padilla R.A., Cadorin D. An innovations approach to dual control // IIIEE Trans. Automat. Control. - 1982. -Vol.27. -Nl. -P.133-137.

75. Cichocki A., Unbehauen R. Neural networks for optimization and signal processing. – New York: John Wiley & Sons, 1998. – 52lp.

76. Garcia C.E., Prett D.M., Morari M. Model predictive control: theory and practice a survey // Automatica. – 1989. – Vol.25. – P.335-348.

77. Recurrent Neural Network-Based Model Predictive Control for Continuous Pharmaceutical Manufacturing (Wong W.C., Chee E., Li J., Wang X.) — *Mathematics*, 2018, 6(11):242.

78. Neural Network Based Model Predictive Control (Piche S., Keeler J.D., Martin G., Boe G., Johnson D., Gerules M.) — *Advances in Neural Information Processing Systems NIPS'99*, 1999.

79. Review on model predictive control: an engineering perspective — *The International Journal of Advanced Manufacturing Technology*, 2021, Vol. 117, pp. 1327-1349.

80. Lan J. *Efficient model predictive control for nonlinear systems modelled by deep neural networks* : e-print (arXiv) / J. Lan. - 2024..

81. Narendra K.S., Parthasarathy K. Identification and control of dynamical system using neural networks // IEEE Trans. Neural Networks.–1990. – Vol.1.–№1. – P.4-27.

82. Leontaritis I.J., Billings S.A. Input-output parametric model for nonlinear systems. Part 1: Deterministic non-linear systems. Part 2: Stochastic non-linear systems // Int. J. Control. – 1985. – Vol.41. – №2. –P. 303-344.

83. Bunke J. Kunstliche neuronale Netze zur Systemidentifikation aus gestorten Mefiwerten. Forschr. Ber. VDI, Reihe 8, № 667. – Dusseldorf: VDI Verlag. – 1997. – P. 37-51.

84. Теорія автоматичного керування : навчальний посібник / П. В. Леонтьєв та ін. ; за заг. ред. П. В. Леонтьєва. – Суми : Сумський державний університет, 2024. – 296 с.

85. Теорія автоматичного управління : навчальний посібник [Електронний ресурс] : навч. посіб. / О. Й. Штіфзон, П. В. Новіков, В. П. Бунь. – Київ : КПІ ім. Ігоря Сікорського, 2020. – 144 с.

86. Теорія автоматичного управління. Нелінійні та дискретні системи [Електронний ресурс] : навчальний посібник / О. Й. Штіфзон, П. В. Новіков. – Київ : КПІ ім. Ігоря Сікорського, 2021. – 98 с.

87. Теорія автоматичного керування : навчальний посібник / П. В. Леонтьєв та ін. ; за заг. ред. П. В. Леонтьєва. – Суми : Сумський державний університет, 2024. – 296 с.

88. Теорія автоматичного керування : навчальний посібник / О. К. Аблесімов – К. : «Освіта України», 2019. – 270 с.

89. Навчальний посібник з дисципліни "Теорія автоматичного керування" : у 2 ч. Ч. 1 / А. П. Гуров, С. І. Ольшевський, О. О. Черно, Л. І. Бугрім. – Миколаїв : НУК, 2018. – 111 с.

90. Теорія автоматичного управління : курс лекцій / Т. М. Боровська. – Вінниця : ВНТУ, 2018. – 256 с.

91. Теорія автоматичного управління: Навчальний посібник [Електронний ресурс] : навч. посіб. для студ. спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології», освітньо-професійна програма «Автоматизація та комп'ютерно-інтегровані технології кібер-енергетичних систем»; уклад.: О. Й. Штіфзон, П. В. Новіков, В.П. Бунь. – Електронні текстові дані (1 файл: 2,2 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2020. – 144 с

92. Лістровий С. В., Мірошник М. А., Клименко Л. А. Теорія автоматичного керування, штучний інтелект і автоматизація процесу прийняття рішення: Навч. посібник. – Харків: УкрДУЗТ, 2019. – 120 с.

93. Автоматизований електропривод. Розділ 1. Теорія електропривода. Навчальний посібник. / Н.Д. Красношапка – К.: КПІ ім. Ігоря Сікорського. - 2023. – 101 с.

94. Возняк, О.М., Штуць. А.А., Колісник М.А. Сучасні системи електроприводів. Теорія та практика. Частина 1. Навчальний посібник. – Вінниця: ТВОРИ, 2021. – 280 с.

95. А. А. Видмиш, Л. В. Ярошенко. Основи електропривода. Теорія та практика. Частина 1. Навчальний посібник. – Вінниця: ВНАУ, 2020. – 387 с.

96. Панкратов А.І. Системи керування електроприводами. Видання 2: Навч. посібник з дисципліни «Системи керування електроприводами» (для студентів спеціальності 151 «Автоматизація та комп'ютерноінтегровані технології» ден-

ної і заочної форми навчання) – Краматорськ: ДДМА, 2018. – 225 с.

97. Колб Ант. А, Колб А. А.Теорія електроприводу: Навчальний посібник. – 2-е вид. перероб. і доп. – Д., Національний гірничий університет, 2011. – 540 с.

98. Панкратов А.І. Системи керування електроприводами. Видання 2: Навч. посібник з дисципліни «Системи керування електроприводами» (для студентів спеціальності 151 «Автоматизація та комп'ютерноінтегровані технології» денної і заочної форми навчання) – Краматорськ: ДДМА, 2018. – 225 с.

99. Василега П. О. Електропривод робочих машин : підручник / П. О. Василега. – Суми: Сумський державний університет, 2022. – 290 с. ISBN 978-966-657-900-6.

100. Шефер О.В. Автоматизований електропривод загальнопромислових механізмів: конспект лекцій. – Полтава: ПолтНТУ, 2020. – 154 с.

101. Електропривод виробничих машин і механізмів: Навчальний посібник / О.Ю. Синявський, В.В. Савченко, В.Я. Бунько, В.Ю. Рамш; За ред. О.Ю. Синявського. – К.: ФОП Ямчинський О.В. , 2020. – 444 с. ISBN 978-617-7890-88-0.

102. Гурко О.Г. Аналіз та синтез систем автоматичного управління у MATLAB: Навчальний посібник / О.Г. Гурко, І.Ф. Єрьоменко. Харків, ХНАДУ, 2012. – 284 с.

Моделювання систем управління в SIMULINK : навч. посібник / В. О. Богомолов, О. Г. Гурко, В. І. Клименко, Д. М.

ДОДАТОК А
БАГАТОШАРОВІ ПРЯМОНАПРАВЛЕНІ МЕРЕЖІ

А.1 Структура штучного нейрона

Основним елементом всіх мереж є нейрон. Штучний нейрон відображає структуру біологічного нейрона (див. рис.А.1). Він має декілька входів і один вихід, який може бути пов'язаний з багатьма входами інших нейронів. Кожен подібний зв'язок має свій коефіцієнт посилення. Нейрон вчиться шляхом модифікації коефіцієнтів посилення відповідно до закону навчання.

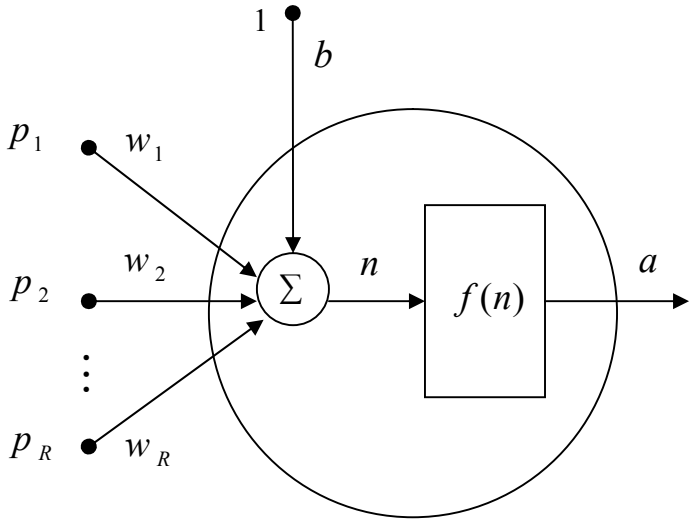
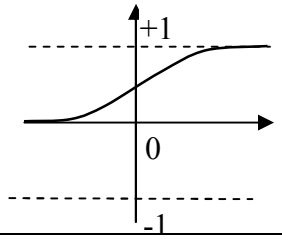
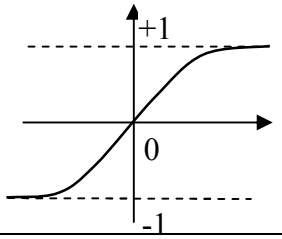
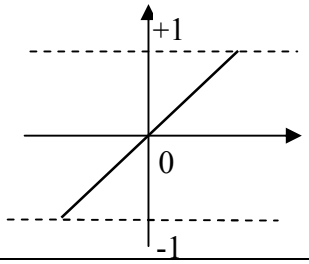
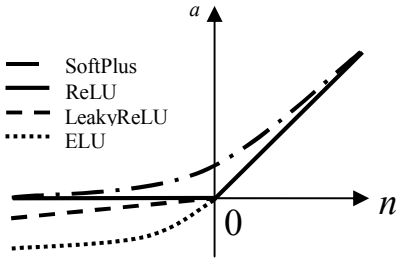


Рисунок А.1 - Схема штучного нейрона

Нейрон реалізує скалярну функцію векторного аргументу. Сигнали обробляються в нейроні таким чином. Кожен вхідний сигнал p_i, умножається на відповідний ваговий коефіцієнт w_i і складається з іншими зваженими вхідними сигналами. Порогова величина, яка позначена як b розглядається в більшості випадків як додатковий ваговий коефіцієнт w_0. Потім отримана сума n перетвориться за допомогою функції активації або передавальної функції f. Як передавальна функція в нейроні може використовуватися принципово будь-яка функція. Вид активаційної функції надає істотний вплив на показники якості регулювання електромеханічної системи при нейроконтролерному управлінні. Для навчання методом зворотного розповсюдження помилки необхідна передавальна функція, що диференціюється. У таблиці А.1 представлені найчастішим використовувані передавальні функції.

Таблиця А.1 – Передавальні функції

Передавальна функція	Графічне зображення	Функція	Похідна
сигмоїдальна		$f(n)=\frac{1}{1+e^{-n}}$	$f'(n)=f(n)(1-f(n))$
тангенс-гіперболічна		$f(n)=\frac{1-e^{-2n}}{1+e^{-2n}}$	$f'(n)=1-f^2(n)$
лінійна		$f(n)=n$	$f'(n)=1$
ReLU і подібні		$f(n)=\begin{cases}0, & n<0\\ n, & n\geq 0\end{cases}$	$f(n)=\begin{cases}0, & n<0\\ 1, & n\geq 0\end{cases}$

Рівняння нейрона має вид

$$a=f\left(\sum_{i=1}^{R} w_i \cdot p_i + b\right).$$

А.2 Одношарові нейронні мережі

Внутрішній потенціал нейронних технологій виявляється при об'єднанні окремих нейронів в нейромережеві структури. З огляду на те, що в даний час існує величезна кількість різних типів мереж, неможливо розглянути кожний з них детально. Тому далі виклад обмежується тільки фактично використовуваними в цій роботі типами мереж. Сукупність нейронів з однаковими вхідними

сигналами називають шаром. Одношарова прямонаправлена мережа зв'язується зазвичай з поняттям одношаровий персептрон. Якщо нейрон має одну з сигмоїдних функцій, то значення його вихідної величини будуть обмежені в деякому діапазоні. Вихід лінійного нейрона може приймати будь-які значення.

В результаті об'єднання декілька нейроних шарів виникає типова багатошарова мережа – багатошаровий персептрон (БП). БП тренується переважно алгоритмом зворотного розповсюдження помилки. Одношарові мережі описується однозначно вказівкою числа входів R і числа нейронів S (число виходів S), і, типом передавальної функції. Розгорнена схема мережі показана на рис.А.2.

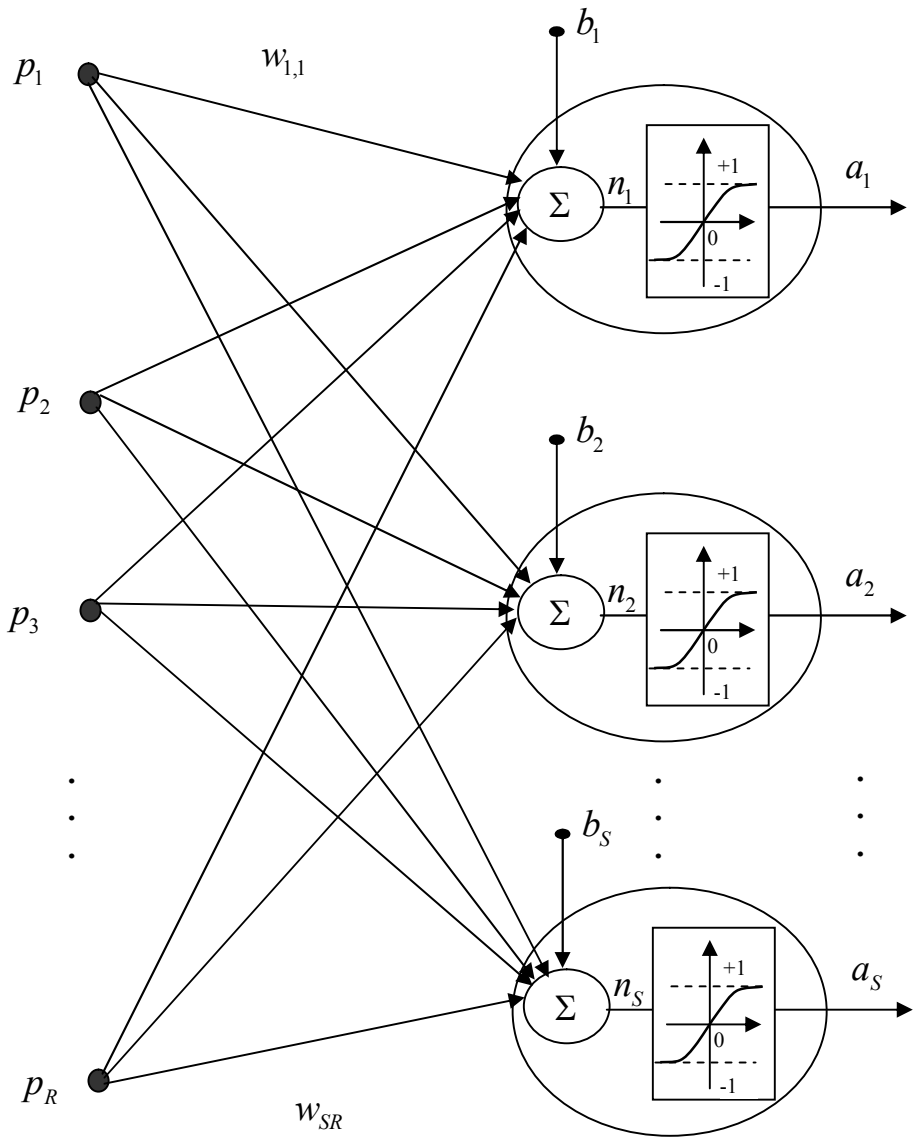


Рисунок А.2 - Структурна схема одношарової мережі (одношаровий персептрон)

В цій мережі кожен елемент вектора входу сполучений зі всіма входами нейрона і це з'єднання задається матрицею вагів ; **W** при цьому кожен i-й нейрон включає елемент, що підсумовує, який формує скалярний вихід $n(i)$. Сукупність скалярних функцій $n(i)$ об'єднується в S-елементний вектор входу **n** функції активації шару. Виходи шару нейронів формують вектор-стовпець **a**, і, таким чином, опис шару нейронів має вигляд:

$$\mathbf{a} = \mathbf{f}(\mathbf{W} \cdot \mathbf{p} + \mathbf{b}).$$

У кожному шарі, як правило, використовується одна і та ж функція активації. Проте можна створювати складені шари нейронів з використанням різних функцій активації, сполучаючи мережі, подібні зображеною на рис.. А.2, паралельно. Обидві мережі матимуть ті ж самі входи, і кожна мережа генеруватиме певну частину виходів. Елементи вектора входу передаються в мережу через матрицю вагів **W**, що має вигляд:

$$\mathbf{W} = \begin{vmatrix} w_{11} & w_{12} & \cdots & w_{1R} \\ w_{21} & w_{22} & \cdots & w_{2R} \\ \vdots & \vdots & \ddots & \vdots \\ w_{S1} & w_{S2} & \cdots & w_{SR} \end{vmatrix}.$$

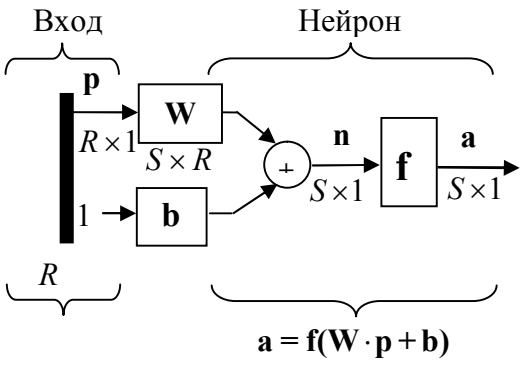


Рисунок А.3 – Укрупнена структурна схема одношарової мережі

Індекси рядків матриці **W** указують адресатів (пункти призначення) вагів нейронів, а індекси стовпців – яке джерело є входом для цієї ваги. Таким чином, елемент матриці вагів $w_{1,2} = \mathbf{W}(1,2)$ визначає коефіцієнт, на який умножається другий елемент входу при передачі його на перший нейрон. Укрупнена структурна схема одношарової мережі показана на рис. А.3.

А.3 Багатошарові нейронні мережі

Для багатошарового персептрона повинні указуватися кількість і тип нейронів всіх прихованих шарів і вихідного шаруючи. Вхідний шар, як правило, не вважається шаром, оскільки сигнали, що поступають в нього, не змінюються. На рис.А.4 представлений БП з R входами, S^1 і S^2 нейронами в прихованих шарах і S^3 нейронами у вихідному шарі. Ця ж тришарова мережа може бути представлена у вигляді укрупненої структурної схеми (рис.А.5).

Перевага цих мереж в тому, що принцип їх роботи описується простими матричними операціями. Для матриць вагів входу і виходу шару використовуються позначення **IW** (Input Weight) і **LW** (Layer Weight) відповідно (такі позначення прийняті в системі MATLAB, яка використовувалася для досліджень в даній роботі). Вихід останнього шару $\mathbf{a}^3$ позначений через $\mathbf{y}$. Це зроблено для того, щоб підкреслити, що вихід останнього шару є виходом мережі.

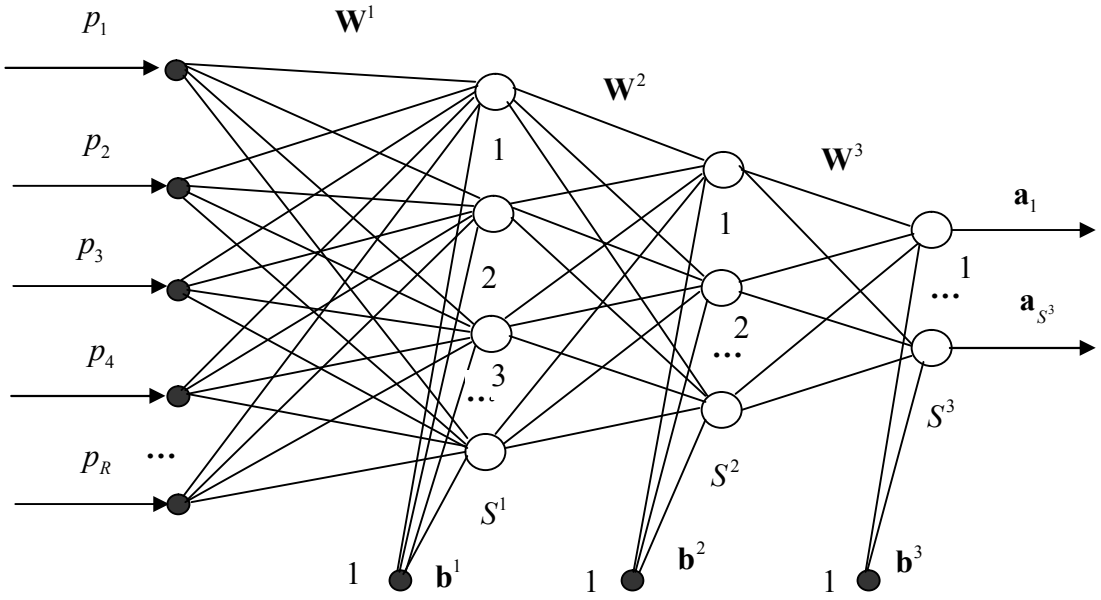


Рисунок А.4 – Структурна схема тришарової мережі
(тришаровий персептрон)

Як правило, для вихідного шару вибирається лінійна передавальна функція, щоб уникнути обмежень вихідної величини в деякому діапазоні. Застосу-

вання декількох наступних один за одним лінійних шарів було б даремним, оскільки їх можна об'єднати без втрати інформації в один шар. Про оптимальний вибір передавальної функції прихованих шарів не існує до цих пір ніяких тверджень. З прагматичних міркувань на практиці призначають в межах шару однакові передавальні функції і найчастіше це сигмоїдні функції. З погляду здібностей апроксимацій не має ніякого значення, яка з двох сигмоїдних функцій вибирається, оскільки одну можна перераховувати в іншу. МП є універсальним апроксиматором-класифікатором і належить в сьогоднішній практиці до найбільш використовуваних нейронних мереж.

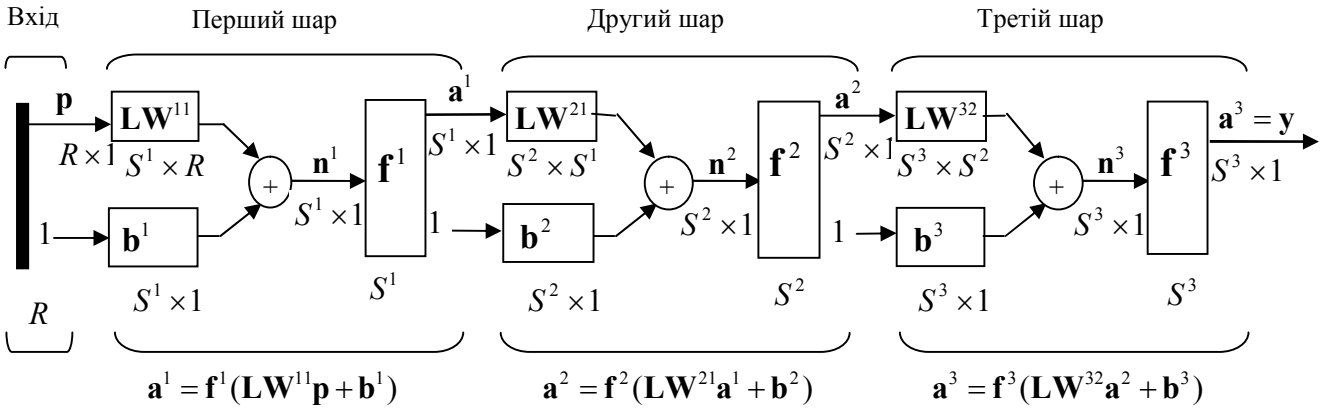


Рисунок А.5 – Укрупнена структурна схема тришарової мережі

А.4 Здібності апроксимації і узагальнення багатошарових мереж

Принципово в більшій частині практичного застосування нейронних мереж йдеться, про те, щоб реалізувати відображення функції за допомогою кінцевого числа параметрів мережі. При цьому повинні виконуватися дві наступні вимоги: апроксимувати представлений для навчання набір даних; узагальнити отримані знання на невідомі вхідні дані.

Виконання першої вимоги означає знайти мережу, яка може з будь-якою точністю вивчити будь-яку задану функцію. Проблема відображення багатовимірної нелінійної функції за допомогою лінійної суперпозиції простих нелінійних функцій сходить до так званої 13-ої проблеми Гілберта. Гілберт припускав,

що ця проблема не має рішення. Це припущення було спростоване теоремою Колмогорова-Арнольдса в кінці 50-х років. В кінці 80-х років ця теорема була переведена на термінологію нейронних мереж Р. Хехт-Нильсоном і Цибенко.

Публікації останніх років зачіпали питання здібностей апроксимацій прямонаправлених нейронних мереж. Для докладного розгляду доказів і деталей слід звернутися до спеціальної літератури. Далі представляються тільки підсумки математичних викладень:

- Теоретично завжди існує така прямонаправлена нейронна мережа з одним нелінійним прихованим шаром і лінійним вихідним шаром, яка здатна відображати будь-які шматково-безперервні багатовимірні функції.
- Необхідна якість апроксимації завжди досягається при кінцевому числі нейронів, якщо задана певна точність апроксимації.

Тільки для небагатьох нейронних структур в сьогодення існує строге математичне обгрунтування їх здібностей. БП в даний момент краще всього досліджений теоретично:

- Збіжність процесу навчання доведена теоремою Perceptron-Konvergenz, яка стверджує, що функція може бути апроксимована за кінцеве, але не відоме, число кроків.
- БП з одним прихованим шаром формує гіперплощині. При двох прихованих шарах утворюються замкнуті гіперпростори, які досить, щоб вирішувати які завгодно по складності завдання класифікації. Подальші шари нічого не покращують, а збільшують час навчання.

Оцінка числа нейронів в прихованих шарах мережі – все ще нетривіальне завдання. Не підлягає сумніву факт вирішального впливу на розмір мережі ступеня нелінійності об'єкту, що ідентифікується. Існують численні емпіричні формули, що дають мінімальне і максимальне число нейронів або зв'язків для забезпечення необхідної точності апроксимації. Але на сьогоднішній день вони дуже неточні. Запропоновані також методи автоматичного підбору числа нейронів в процесі тренування, які можна розділити на дві групи: група редукуючих методів і група конструюючих методів. Редукуючі алгоритми менш ефек-

тивні, оскільки вони мають, принаймні, два недоліки: надмірне число нейронів при ініціалізації мережі невідоме і тому оцінюється приблизно, із-за явно надмірного числа нейронів процес тренування протікає досить поволі. Сьогоднішня практика свідчить, що систематичний підбір оптимального числа нейронів також є ефективним. З вищесказаного виходить, що для успішної ідентифікації деякої нелінійної системи мають значення вибір типу мережі, виду функції активації, числа нейронів (а для БП ще і числа прихованих шарів).

Здібності нейронних мереж до узагальнення дозволяють проводити тренування при одних парах тренувальних даних і після цього отримувати правильний вихід при інших можливих вхідних даних. Виникаюча проблема формування інформативних тренувальних даних не вирішена в загальному вигляді до цих пір. Проте, успішно практикується наступний емпіричний метод. Узагальнювані вхідні дані не повинні сильно відрізнятися від тренувальних даних. Це вимушує обмежуватися при навчанні небагатьма крапками даних, між якими мережею виконуватиметься нелінійна інтерполяція.

ДОДАТОК Б
НАВЧАННЯ НЕЙРОННИХ МЕРЕЖ

Б.1 Завдання навчання

При рішенні за допомогою нейронних мереж прикладних завдань необхідно зібрати точний і представницький об'єм даних для того, щоб навчити нейронну мережу рішенню таких задач. Повчальний набір даних – це набір спостережень, що містять ознаки об'єкту, що вивчається. Перше питання, які ознаки використовувати і скільки і які спостереження треба провести.

Вибір ознак, принаймні первинний, здійснюється евристично на основі наявного досвіду, який може підказати, які ознаки є найбільш важливими. Спочатку слід включити всі ознаки, які, на думку аналітиків або експертів, є істотними, на подальших етапах ця множина буде скорочена.

Нейронні мережі працюють з числовими даними, узятими, як правило, з деякого обмеженого діапазону. Це може створити проблеми, якщо значення спостережень виходять за межі цього діапазону або пропущені.

Питання про те, скільки потрібно мати спостережень для навчання мережі, часто виявляється непростим. Відомий ряд евристичних правил, які встановлюють зв'язок між кількістю необхідних спостережень і розмірами мережі. Просте з них свідчить, що кількість спостережень повинна бути в 10 разів більше числа зв'язків в мережі. Насправді це число залежить від складності того відображення, яке повинна відтворювати нейронна мережа. Із зростанням числа використовуваних ознак кількість спостережень зростає по нелінійному закону, так що вже при досить невеликому числі ознак, скажемо 50, може потрібно величезне число спостережень. Ця проблема носить назву «прокляття розмірності».

Для більшості реальних завдань буває достатнім декількох сотень або тисяч спостережень. Для складних завдань може потрібно більша кількість, проте дуже рідко зустрічаються завдання, де потрібно менше 100 спостережень. Якщо

даних мало, то мережа не має достатньої інформації для навчання, і краще, що можна в цьому випадку зробити, – це спробувати підігнати до даним деяку лінійну модель.

Б.2 Процедури адаптації і навчання

Після того, як визначена кількість шарів мережі і число нейронів в кожному з них, потрібно призначити значення вагів і зсувів, які мінімізують помилку рішення. Це досягається за допомогою процедур навчання. Шляхом аналізу тих, що є у розпорядженні аналітика вхідних і вихідних даних ваги і зсуву мережі автоматично настроюються так, щоб мінімізувати різницю між бажаним сигналом і отриманим на виході в результаті моделювання. Ця різниця носить назву помилки навчання. Таким чином, процес навчання – це процес підгонки параметрів тієї моделі процесу або явища, яка реалізується нейронною мережею. Помилка навчання для конкретної конфігурації нейронної мережі визначається шляхом прогону через мережу всіх наявних спостережень і порівняння вихідних значень з бажаними, цільовими значеннями. Ці різниці дозволяють сформувати так звану функцію помилок (критерій якості навчання). Як така функція найчастіше береться сума квадратів помилок. При моделюванні нейронних мереж з лінійними функціями активації нейронів можна побудувати алгоритм, що гарантує досягнення абсолютного мінімум помилки навчання. Для нейронних мереж з нелінійними функціями активації в загальному випадку не можна гарантувати досягнення глобального мінімуму функції помилки.

При такому підході до процедури навчання може виявитися корисним геометричний аналіз поверхні функції помилок. Визначимо ваги і зсуву як вільні параметри моделі і їх загальне число позначимо через N; кожному набору таких параметрів поставимо у відповідність одне вимірювання у вигляді помилки мережі. Тоді для всіляких поєднань вагів і зсувів відповідну помилку мережі можна зобразити точкою в N+1-вимірному просторі, а всі такі точки утворюють деяку поверхню, звану поверхнею функції помилок. При такому підході мета

навчання нейронної мережі полягає в тому, щоб знайти на цій багатовимірній поверхні глобальний мінімум.

У разі лінійної моделі мережі і функції помилок у вигляді суми квадратів така поверхня буде параболоїдом, який має єдиний мінімум і це дозволяє відшукати такий мінімум досить просто.

У разі нелінійної моделі поверхня помилок має набагато складнішу будову і має ряд несприятливих властивостей, зокрема може мати локальні мінімуми, плоскі ділянки, сідлові точки і довгі вузькі яри.

Визначити глобальний мінімум багатовимірної функції аналітично неможливо, і тому навчання нейронної мережі, по суті справи, є процедурою вивчення поверхні функції помилок. Відштовхуючись від випадково вибраної точки на поверхні функції помилок, алгоритм навчання поступово відшукує глобальний мінімум. Як правило, для цього обчислюється градієнт (нахил) функції помилок в даній точці, а потім ця інформація використовується для просування вниз по схилу. Врешті-решт алгоритм зупиняється в деякому мінімумі, який може опинитися лише локальним мінімумом, а якщо повезе, то і глобальним.

Таким чином, по суті алгоритми навчання нейронних мереж аналогічні алгоритмам пошуку глобального екстремуму функції багатьох змінних. Серед останніх слід виділити алгоритми зв'язаних градієнтів і Левенберга-Марквардта (Levenberg-Marquardt).

Проте з урахуванням специфіки нейронних мереж для них розроблені спеціальні алгоритми навчання, серед яких слід виділити алгоритм зворотного розповсюдження помилки .

При використанні алгоритму зворотного розповсюдження помилки мережа розраховує помилку, що виникає у вихідному шарі, і обчислює вектор градієнта як функцію вагів і зсувів. Цей вектор указує напрям найкоротшого спуску по поверхні для даної крапки, тому якщо просунутися в цьому напрямі, то помилка зменшиться. Послідовність таких кроків врешті-решт приведе до мінімуму того або іншого типу. Певну трудність тут викликає вибір величини кроку.

При великій довжині кроку збіжність буде швидшою, але є небезпека перестрибнути через рішення або піти в неправильному напрямі. Класичним прикладом такого явища при навчанні нейронної мережі є ситуація, коли алгоритм дуже поволі просувається по вузькому яру з крутими схилами, перестрибуючи з одного схилу на інший. Навпаки, при малому кроці, ймовірно, буде вибрано вірний напрям, проте при цьому буде потрібно дуже багато ітерацій. На практиці величина кроку вибирається пропорційній крутизні схилу (градієнту функції помилок); такий коефіцієнт пропорційності називається параметром швидкості настройки. Правильний вибір параметра швидкості настройки залежить від конкретного завдання і зазвичай здійснюється досвідченим шляхом; цей параметр може також залежати від часу, зменшуючись у міру виконання алгоритму.

Алгоритм діє ітеративно і його кроки прийнято називати епохами або циклами. На кожному циклі на вхід мережі послідовно подаються всі повчальні спостереження, вихідні значення порівнюються з цільовими значеннями і обчислюється функція помилки. Значення функції помилки, а також її градієнта використовуються для коректування вагів і зсувів, після чого всі дії повторюються. Початкові значення вагів і зсувів мережі вибираються випадковим чином, і процес навчання припиняється або коли реалізована певна кількість циклів, або коли помилка досягне деякого малого значення або перестане зменшуватися.

Б.3 Явище перенавчання

Одна з найбільш серйозних труднощів при навчанні мережі полягає в тому, що у ряді випадків ми мінімізуємо не ту помилку, яку насправді потрібно мінімізувати; потрібно мінімізувати помилку, яка з'являється в мережі, коли на неї подаються абсолютно нові спостереження. Вельми важливо, щоб нейронна мережа володіла здатністю пристосовуватися до цих нових спостережень. Що ж відбувається насправді? Мережа навчається мінімізувати помилку на деякій обмеженій повчальній множині. Це не відповідає вимогам теорії про наявність ідеальної і нескінченно великої повчальної множини. І це не відповідає тій реа-

льній ситуації, коли треба мінімізувати конкретну функцію помилок для наперед невідомої моделі.

Це породжує проблему, яка відома як явище перенавчання. Звернемося до завдання апроксимації деякої функції многочленом. Графіки многочленів часто мають вельми хитромудрі форми, і чим вище ступінь многочлена, тим складніше їх форма. Якщо є деякий набір даних, то можна поставити мету підібрати для нього апроксимуючий многочлен і таким чином отримати відповідну математичну модель для цього набору даних. Оскільки початкові дані, як правило, задані з погрішностями, то не можна вважати, що краща модель задається кривою, яка проходить точно через задані точки. Многочлен низького порядку може виявитися достатньо грубим для апроксимації даних, тоді як многочлен високого порядку може точно слідувати даним, приймаючи при цьому вельми хитромудру форму, що не має ніякого відношення до форми дійсної залежності. Остання ситуація і демонструє те, що називається явищем перенавчання.

При роботі з нейронними мережами користувач стикається з тією ж проблемою. Мережі з великою кількістю вагів дозволяють відтворювати дуже складні функції, і в цьому сенсі вони схильні до перенавчання. Мережа ж з невеликою кількістю вагів може опинитися недостатньо гнучкою, щоб змоделювати наявну залежність. Наприклад, одношарова лінійна мережа здатна відтворювати тільки лінійні функції. Якщо використовувати багатошарові лінійні мережі, то помилка завжди буде менша, але може свідчити не про хорошу якість моделі, а про те, що виявляється явище перенавчання.

Для того, щоб виявити ефект перенавчання, використовується механізм контрольної перевірки. Частина повчальних спостережень резервується як контрольні спостереження і не використовується при навчанні мережі. Натомість у міру роботи алгоритму ці спостереження застосовуються для незалежного контролю результату. Спочатку помилка мережі на повчальне і контрольному множинах буде однаковою; якщо вони істотно відрізняються, то, мабуть, це означає, що розбиття спостережень на 2 множини не забезпечило їх однорідність. У міру навчання мережі помилка убуває, і, поки навчання зменшує функ-

цію помилок помилка на контрольній множині також убуватиме. Якщо ж контрольна помилка перестала убувати або стала рости, це указує на те, що мережа почала дуже близько слідувати початковим даним і навчання слід зупинити. В цьому випадку слід зменшити кількість нейронів або шарів, бо мережа є дуже могутнього для вирішення даного завдання Якщо ж, навпаки, мережа має недостатню потужність, щоб відтворити наявну залежність, те явище перенавчання швидше за все спостерігатися не буде і обидві помилки – навчання і перевірки – не досягнуть достатньо малого рівня.

Що виникають при роботі з нейронними мережами проблеми відшукання глобального мінімуму або вибору розміру мережі призводять до того, що при практичній роботі доводиться експериментувати з великим числом мереж різних конфігурацій, деколи навчаючи кожну з них кілька разів і порівнюючи отримані результати. Головним критерієм вибору в цих випадках є контрольна погрішність. При цьому застосовується правило, згідно якому з двох нейронних мереж з приблизно рівними контрольними погрішностями слід вибирати ту, яка простіше.

Необхідність багатократних експериментів веде до того, що контрольна множина починає грати ключову роль у виборі моделі нейронної мережі, тобто стає частиною процесу навчання. Тим самим його роль як незалежного критерію якості моделі ослабляється, оскільки при великому числі експериментів виникає ризик перенавчання нейронної мережі на контрольній множині. Для того, щоб гарантувати надійність вибираної моделі мережі, резервують ще одне – тестова безліч спостережень. Підсумкова модель тестується на даних з цієї множини, щоб переконатися, що результати, досягнуті на повчальній і контрольній множинах реальні. Зрозуміло, для того, щоб добре грати свою роль, тестова множина повинна бути використана тільки 1 раз: якщо його використовувати повторно для коректування процесу навчання, то воно фактично перетвориться на контрольну множину.

Отже, процедура побудови нейронної мережі складається з наступних кроків:

вибору початкової конфігурації мережі; наприклад, у вигляді одного шару з числом нейронів, рівним 1/2 загальної кількості входів і виходів;

моделювання і навчання мережі з оцінкою контрольної помилки і використанням додаткових нейронів або проміжних шарів;

виявлення ефекту перенавчання і коректування конфігурації мережі.

Б.4 Властивість узагальнення

При описі процедури навчання нейронних мереж неявно використовувалося припущення, що повчальна, контрольна і тестова множини є представницькими для вирішуваної задачі. Зазвичай як повчальні беруться дані, випробувань на ряду прикладів. Якщо обставини змінилися, то закономірності, що мали місце у минулому, можуть більше не діяти.

Крім того, нейронна мережа може навчатися тільки на тих даних, які вона має в своєму розпорядженні. Припустимо, що відоме повчальна множина для системи стабілізації літака при польоті в спокійній атмосфері, а потрібно спроектувати систему стабілізації на основі нейронної мережі для умов польоту при сильних збуреннях. Тоді навряд чи можна чекати від мережі правильного рішення в абсолютно новій для неї ситуації.

Класичним прикладом непредставницької моделі нейронної мережі є наступна ситуація. При проектуванні системи машинного зору, призначеної для автоматичного розпізнавання цілей, мережа навчалася на 100 картинках, що містять зображення танків, і на 100 інших картинках, де танків не було. Після навчання мережі був досягнутий стовідсотково «правильний» результат. Але коли на вхід мережі були подані нові дані, вона безнадійно провалилася. У чому ж була причина? З'ясувалося, що фотографії з танками були зроблені в похмурий, дощовий день, а фотографії без танків – в сонячний день. Мережа навчилася уловлювати різницю в загальній освітленості. Щоб мережа могла результативно працювати, її слід було навчати на даних, де б були присутні всі погодні

умови і типи освітлення, при яких мережу передбачається використовувати, і це не кажучи ще про рельєф місцевості, вугілля і дистанцію зйомки і т.д.

Якщо мережа мінімізує загальну погрішність, великого значення набувають пропорції, в яких представлені дані різних типів. Мережа, навчена на 900 «хороших» і 100 «поганих» спостереженнях, спотворюватиме результат на користь хороших спостережень, оскільки це дозволить алгоритму зменшити загальну погрішність. Якщо в реальній ситуації «хороші» і «погані» об'єкти представлені в іншій пропорції, то результати, що видаються мережею, можуть виявитися невірними. Прикладом цього може бути завдання виявлення захворювань. Хай, наприклад, при звичайних обстеженнях в середньому 90% людей виявляються здоровими і мережа, таким чином, навчається на даних, в яких пропорція здорові/хворі рівна 90/10. Потім ця ж мережа застосовується для діагностики пацієнтів з певними скаргами, серед яких співвідношення здорові/хворі вже 50/50. В цьому випадку мережа ставитиме діагноз занадто обережно і не розпізнаватиме захворювання у деяких хворих. Якщо ж, навпаки, мережу навчити на даних «з скаргами», а потім протестувати на «звичайних» даних, то вона видаватиме підвищене число неправильних діагнозів про наявність захворювання. У таких ситуаціях повчальні дані потрібно скоректувати так, щоб були враховані відмінності в розподілі даних (наприклад, можна повторити рідкісні спостереження або видалити що часто зустрічаються). Як правило, краще всього постаратися зробити так, щоб спостереження різних типів були представлені рівномірно, і відповідно цьому інтерпретувати результати, які видає мережа.

Здатність мережі, навченої на деякій безлічі даних, видавати правильні результати для достатньо широкого класу нових даних, у тому числі і не представлених при навчанні, називається властивістю узагальнення нейронної мережі.

Інший підхід до процедури навчання мережі можна сформулювати, якщо розглядати її як процедуру, зворотну моделюванню. В цьому випадку потрібно підібрати такі значення вагів і зсувів, які забезпечували б потрібну відповід-

ність між входами і бажаними значеннями на виході. Така процедура навчання носить назву адаптації і достатньо широко застосовується для настройки параметрів нейронних мереж.

Б.5 Вибір методу навчання нейронної мережі

Б.5.1 Критерій якості навчання. Методи навчання.

Фаза навчання є найбільш проблематичною при використанні нейронних мереж. Для БП мереж процес навчання є багатозмінним нелінійним оптимізаційним завданням. Її аналітичне рішення неможливе, що вимагає використання ітераційних методів оптимізації.

У загальному випадку початковими даними є: структура мережі, тренувальні дані і критерій оцінки якості тренування. Мета оптимізації – визначити параметри мережі, що забезпечують виконання критерію якості. Таким критерієм якості найчастіше виступає мінімум квадратичного функціонала:

$$J = \frac{1}{2}\sum_{q=1}^{Q}\sum_{i=1}^{S^L}(t_i^q - a_i^{qS^L})^2 \ , \qquad (Б.1)$$

де J – функціонал;

Q – об'єм вибірки;

L – число шарів мережі;

q – номер вибірки;

S^L – число нейронів вихідного шаруючи;

$\mathbf{a}^q = [a_i^{qL}]$ – вектор сигналу на виході мережі;

$\mathbf{t}^q = [t_i^q]$ – вектор бажаних (цільових) значень сигналу на виході мережі для вибірки з номером q.

Потім за допомогою того або іншого методу навчання визначаються значення параметрів (вагів і зсувів), що настроюються, мережі, які забезпечують мінімальне значення функціонала помилки.

Методика тренування нейронних мереж була в основному запозичена з відомих методів теорії оптимізації. Зважаючи на величезне число створених алгоритмів тут можна дати лише їх короткий огляд. Вживані методи навчання грубо можна розділити на наступні основні групи: методи глобальної оптимізації; градієнтні методи першого і другого порядку.

Істотною проблемою, що виникає при використанні градієнтних методів настройки параметрів ШНМ, є їх локальність. В той же час цільова функція (сумарна помилка на тренувальному наборі шаблонів або кумулятивна оцінка ефективності роботи навчаної системи) не унімодальна. Кількість локального оптимуму для більшості практичних завдань навчання обчислюється мільйонами при розмірності пошукового простору порядку $10^2 - 10^3$. Внаслідок цього результат навчання залежить від правильності вибору стартової крапки, і виникає необхідність багатократного повторення процедури настройки параметрів ШНМ. Перераховані проблеми можуть бути вирішені при використанні методів глобальної оптимізації. Найбільш ефективним з них є генетичний алгоритм (ГА). Розглядаючи ШНМ як єдиний набір параметрів, ГА здатний здійснювати її оптимальну настройку при розмірності пошукового простору достатньою для вирішення більшості практичних завдань.

У останні десятиліття розроблена безліч способів навчання ШНМ за допомогою ГА. Отримані результати доводять великі можливості такого симбіозу. Сумісне використання ШНМ і ГА алгоритмів має і ідеологічну перевагу тому, що вони відносяться до методів еволюційного моделювання і розвиваються в рамках одного підходу запозичення технікою природних методів і механізмів як найбільш оптимальних.

Слід зазначити, що завдання глобальної оптимізації вирішуються шляхом систематичного перебору всіх значень всіх аргументів, тому із-за експоненціального зростання складності завдання із зростанням її розмірності практичне застосування цих методів для великих мереж дуже утруднено.

У сучасних завданнях використовуються в основному градієнтні методи навчання. Вони відносно швидко знаходять оптимум. Проте при цьому відсут-

ній доказ того, що знайдений абсолютний екстремум. Численний досвід тренування мереж свідчить, що застосування методів локальної оптимізації сумісне з «виштовхуванням» мережі з локального мінімуму і послідовним збільшенням числа нейронів також приводить до успішного навчання мережі. У справжній роботі використовуються градієнтні методи навчання, тому розглянемо їх детальніше.

Б.5.2 Метод зворотного розповсюдження помилки.

Архітектура багатошарової мережі істотно залежить від вирішуваної задачі. Для лінійних нейронних мереж може бути встановлений зв'язок між сумарною кількістю вагів і зсувів з довжиною повчальної послідовності. Для інших типів мереж число шарів і нейронів в шарі часто визначається досвідом, інтуїцією проектувальника і евристичними правилами.

Навчання мережі включає декілька кроків:

• вибір початкової конфігурації мережі з використанням, наприклад, наступного евристичного правила: кількість нейронів проміжного шару визначається половиною сумарної кількості входів і виходів;

• проведення ряду експериментів з різними конфігураціями мережі і вибір тій, яка дає мінімальне значення функціонала помилки;

• якщо якість навчання недостатньо, слід збільшити число нейронів шаруючи або кількість шарів;

• якщо спостерігається явище перенавчання, слід зменшити число нейронів в шарі або видалити один або декілька шарів.

Нейронні мережі, призначені для вирішення практичних завдань, можуть містити до декількох тисяч параметрів, що настроюються, тому обчислення градієнта може зажадати вельми великих витрат обчислювальних ресурсів. З урахуванням специфіки багатошарових нейронних мереж для них розроблені спеціальні методи розрахунку градієнта, серед яких слід виділити метод зворотного розповсюдження помилки (ЗР), який був розроблений в 1986г. Потім з'явилися численні роботи по прискоренню збіжності алгоритму. Метод зворо-

тного розповсюдження помилки є першим теоретично доведеним алгоритмом для настройки вагів при навчанні багатошарового персептрона. В даний час ЗР може розглядатися як еталонний тест, з яким порівнюються всі інші методи навчання.

Термін «зворотне розповсюдження» відноситься до процесу, за допомогою якого можуть бути обчислені похідні функціонала помилки по параметрах мережі. Цей процес може використовуватися у поєднанні з різними стратегіями оптимізації. Існує багато варіантів алгоритму зворотного розповсюдження. Звернемося до одного з них.

Розглянемо вираз для градієнта критерію якості по вагових коефіцієнтах для вихідного шару L:

$$\frac{dJ}{dw_{ij}^{L}} = \frac{\partial}{dw_{ij}^{L}}\left(\frac{1}{2}\sum_{q=1}^{Q}\sum_{k=1}^{S^M}(t_k^q - a_k^{qM})^2\right) = \sum_{q=1}^{Q}\sum_{k=1}^{S^M}(t_k^q - a_k^{qL})\frac{\partial a_k^{qL}}{dw_{ij}^{K}}\ ;\quad i=\overline{1,S^L};\quad j=\overline{0,S^{L-1}},$$

де S^L – число нейронів в шарі;

a_k^{qL} – k–й елемент вектора виходу шаруючи L для елементу вибірки з номером q.

Правило функціонування шару L записується так:

$$a_k^{qL} = f_L\left(\sum_{i=0}^{S^{L-1}} w_{ij}^{L} a_k^{q(L-1)}\right),\quad k=\overline{1,S^L}\quad .$$

Приймаючи до уваги, що

$$\frac{\partial a_k^{qL}}{\partial w_{ij}^{L}} = \begin{cases} 0, & k\neq i \\ f_L'(n_i^{qL})a_j^{q(L-1)}, & k=i \end{cases} \qquad i=\overline{1,S^L};\quad j=\overline{0,S^{L-1}},$$

отримаємо

$$\frac{dJ}{dw_{ij}^{L}} = -\sum_{q=1}^{Q}(t_i^q - a_i^{qL})f_L'(n_i^{qL})a_i^{q(L-1)}\ .$$

Ввівши позначення для локальної помилки вихідного шару

$$\Delta_i^{qL} = (t_i^q - a_i^{qL}) f_L'(n_i^{qL}), \quad i = 1,...,S^L,$$

отримаємо

$$\frac{dJ}{dw_{ij}^L} = -\sum_{q=1}^{Q} \Delta_i^{qL} a_i^{q(L-1)}; \qquad i = \overline{1, S^M}; \quad j = \overline{0, S^{M-1}}.$$

Перейдемо шару $L-1$ і запишемо співвідношення для настройки вагів w_{ij}^{L-1}

$$\frac{dJ}{dw_{ij}^{L-1}} = -\sum_{q=1}^{Q}\sum_{k=1}^{S^L} (t_k^q - a_k^{qM}) f_L'(n_k^{qL}) \frac{\partial n_k^{qL}}{da_i^{q(L-1)}} \frac{\partial a_k^{q(L-1)}}{dw_{ij}^{L-1}} \; a_i^{q(L-1)} =$$

$$= -\sum_{q=1}^{Q}\sum_{k=1}^{S^L} (t_k^q - a_k^{qL}) f_L'(n_k^{qM}) w_k^L \frac{\partial n_k^{qL}}{da_i^{q(L-1)}} f_{L-1}'(n_i^{q(L-1)}) \; a_j^{q(L-2)} =$$

$$= -\sum_{q=1}^{Q} \Delta_i^{q(L-1)} a_j^{q(L-2)},$$

де Δ_i^{qL} – локальна помилка шару $L-1$

$$\Delta_i^{qL} = \sum_{k=1}^{S^M} (t_k^q - a_k^{qM}) f_M'(n_k^{qM}) w_k^M f_{M-1}'(n_i^{q(M-1)}) = \left(\sum_{k=1}^{S^L} \Delta_k^{qL} \right) f_{L-1}'(n_i^{q(L-1)}), \quad i = \overline{1, S^{L-1}}.$$

Для шарів $L-2$, $L-3$,…,1 обчислення приватних похідних критерію J по елементах матриць вагових коефіцієнтів виконується аналогічно. У результаті отримуємо наступну загальну формулу:

$$\frac{dJ}{dw_{ij}^r} = \sum_{q=1}^{Q} \Delta_i^{q(r-1)} a_j^{q(r-1)}), \quad r = \overline{1, L}, \quad i = \overline{1, S^r}, \quad i = \overline{0, S^{r-1}}, \qquad \text{(Б.2)}$$

де r – номер шару

$$\Delta_i^{qr} = \left(\sum_{k=1}^{S^{r-1}} \Delta_k^{q(r-1)} w_{ki}^{r+1} \right) f_r'(n_i^{qr}), \quad r = \overline{1, M-1};$$

$$\Delta_i^{qL} = (t_i^q - a_i^{qL}) f_L'(n_i^{qL}) \ , \quad i = \overline{1, S^{L-1}} \ .$$

На рис. Б.1 представлена схема обчислень, відповідна виразу (Б.2).

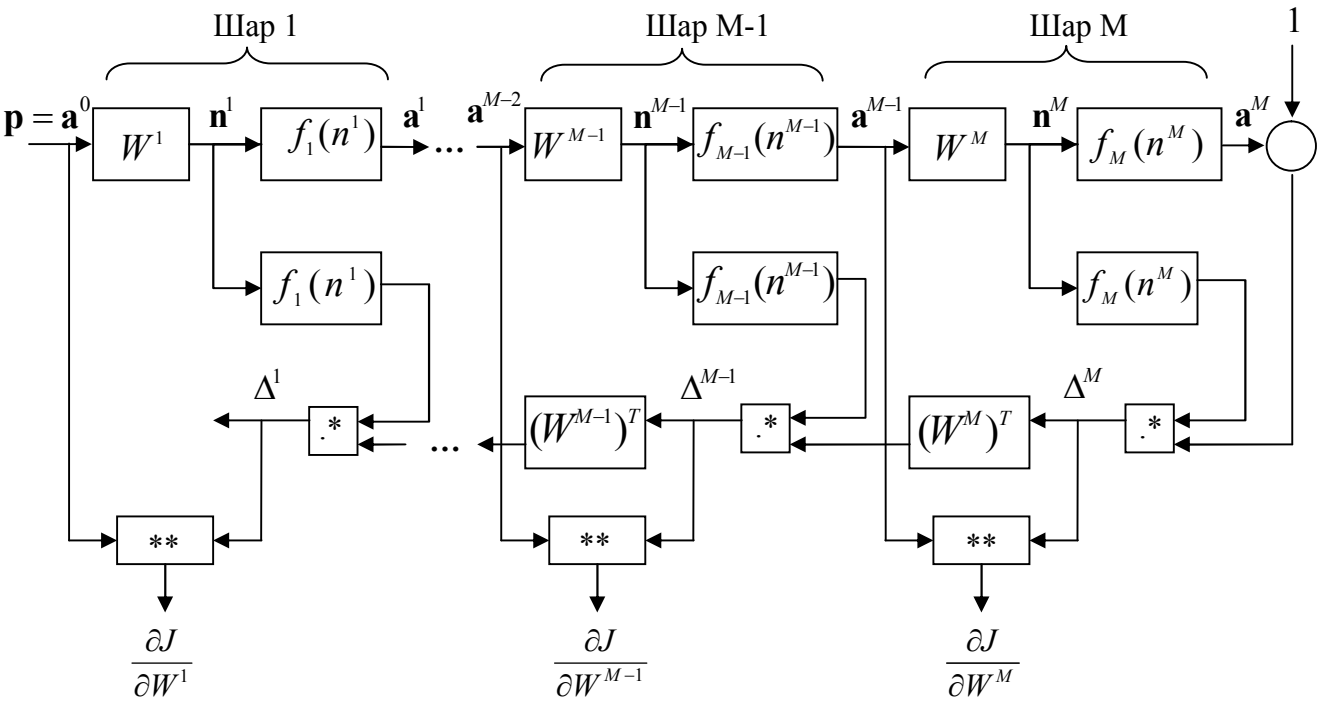


Рисунок Б.1 – Схема обчислення приватних похідних критерію J по елементах матриць вагових коефіцієнтів

На цій схемі символом * позначена операція поелементного множення векторів, а символом ** – множення вектора Δ на $\mathbf{a}^{\mathrm{T}}$; символ, що позначає номер елементу вибірки, скорочено опущений.

Б.5.3 Настройка вагових коефіцієнтів мережі.

Методи, використовувані при навчанні нейронних мереж, багато в чому аналогічні методам визначення екстремуму функції декілька змінних. Вони діляться на дві категорії – методи першого і другого порядку.

У методах *першого* порядку використовується градієнт функціонала по параметрах, що настроюються, і на кожній ітерації k уточнення вектора вагів $\mathbf{w}_k$ проводиться по формулі

$$\mathbf{w}_{k+1} = \mathbf{w}_k - a_k \mathbf{g}_k \, , \qquad \text{(Б.3)}$$

де a_k – параметр швидкості навчання;

$\mathbf{g}_k$ – градієнт функціонала, відповідний ітерації з номером k.

Вектор в напрямі, протилежному градієнту, указує напрям найкоротшого спуску по поверхні функціонала помилки. Якщо реалізується рух в цьому напрямі, то помилка зменшуватиметься. Послідовність таких кроків врешті-решт приведе до значень параметрів, що настроюються, що забезпечують мінімум функціонала. Певну трудність тут викликає вибір параметра швидкості навчання a_k. При великому значенні параметра a_k збіжність буде швидкою, але існує небезпека пропустити рішення або піти в неправильному напрямі. Класичним прикладом є ситуація, коли алгоритм дуже поволі просувається по вузькому яру з крутими схилами, перестрибуючи з одного на інший. Навпаки, при малому кроці, ймовірно, буде вибрано вірний напрям, проте при цьому буде потрібно дуже багато ітерацій. Залежно від прийнятого алгоритму параметр швидкості навчання може бути постійним або змінним. Правильний вибір цього параметра залежить від конкретного завдання і зазвичай здійснюється досвідченим шляхом; у разі змінного параметра його значення зменшується у міру наближення до мінімуму функціонала. Згідно (Б.3) уточнюються значення вагів $\mathbf{w}_k$ в методі ЗР в його стандартній реалізації.

У алгоритмах зв'язаного градієнта пошук мінімуму виконується уздовж зв'язаних напрямів, що забезпечує зазвичай швидшу збіжність, ніж при найшвидшому спуску. Всі алгоритми зв'язаних градієнтів на першій ітерації починають рух у напрямі антиградієнта

$$\mathbf{p}_0 = -\mathbf{g}_0 .$$

Тоді напрям наступного руху визначається так, щоб він був зв'язаний з попереднім. Відповідний вираз для нового напряму руху є комбінацією нового напряму найшвидшого спуску і попереднього напряму:

$$\mathbf{p}_k = -\mathbf{g}_k + \beta_k \cdot \mathbf{p}_{k-1} .$$

де $\mathbf{p}_k$. – напрям руху;

$\mathbf{g}_k$ – градієнт функціонала помилки;

β_k – коефіцієнт відповідають ітерації з номером k.

Коли напрям спуску визначений, то нове значення вектора параметрів, що настроюються, $\mathbf{w}_{k+1}$ обчислюється за формулою

$$\mathbf{w}_{k+1} = \mathbf{w}_k - a_k \cdot \mathbf{p}_k .$$

Градієнтні методи першого порядку характеризуються малою швидкістю збіжності. Це породило помилкову думку про те, що всі нейронні мережі вимагають для тренування великих витрат часу.

Потужніші градієнтні методи *другого* порядку базуються на методі локальної оптимізації Ньютона, який при пошуку оптимуму додатково враховує ще і вигин поверхні функціонала якості (другі похідні).

Основний крок методу Ньютона визначається по формулі

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mathbf{H}_k^{-1}\mathbf{g}_k ,$$

де $\mathbf{H}$ – матриця Гессе других похідних функціонала якості по вагових коефіцієнтах мережі.

У багатьох випадках метод Ньютона сходиться швидше, ніж методи зв'язаного градієнта, але вимагає великих витрат із-за обчислення гессиана. Для того, щоб уникнути обчислення матриці Гессе, пропонуються різні способи її заміни наближеними виразами, що породжує так звані квазіньютонови алгоритми такі як алгоритм, запропонований Бройлером, Флетчером, Гольдфарбом і Шанно, однокроковий алгоритм методу січних площин OSS, алгоритм LM Левенберга-Марквардта і ін.

У даній роботі використовуємо алгоритм Левенберга-Марквардта. Даний алгоритм реалізує наступну стратегію для оцінки матриці Гессе. Оскільки функціонал якості визначається як сума квадратів помилок, то гессиан може бути приблизно обчислений як

$$\mathbf{H} \cong \mathbf{J}^T\mathbf{J} \ , \qquad \text{(Б.4)}$$

а градієнт розрахований по формулі

$$\mathbf{g} \cong \mathbf{J}^T \mathbf{e},$$

де $\mathbf{J} = \dfrac{\partial J}{\partial \mathbf{w}}$ – матриця Якобі похідних функціонала помилки по параметрах, що настроюються;

$\mathbf{e}$ – вектор помилки мережі.

Матриця Якобі може бути обчислена на основі стандартного методу зворотного розповсюдження помилки, що істотно простіше за матрицю Гессе.

У алгоритмі Левенберга-Марквардта нове значення вектора параметрів $\mathbf{w}_{k+1}$, що настроюються, обчислюється за формулою

$$\mathbf{w}_{k+1} = \mathbf{w}_k - (\mathbf{J}^T \mathbf{J} + \mu \mathbf{I})^{-1} \mathbf{J}^T \mathbf{e}_k \ .$$

Коли коефіцієнт μ рівний 0, отримуємо метод Ньютона з наближенням Гессиана у формі (Б.4), коли значення μ велике, отримуємо метод градієнтного спуску з маленьким кроком. Оскільки метод Ньютона має велику точність і швидкість збіжності поблизу мінімуму, завдання полягає в тому, щоб в процесі мінімізації щонайшвидше перейти до методу Ньютона. З цією метою параметр μ зменшують після кожної успішної ітерації і збільшують тільки тоді, коли пробний крок показує, що функціонал помилки зростає. Така стратегія забезпечує зменшення помилки після кожної ітерації алгоритму.

ДОДАТОК В
ЗАСТОСУВАННЯ НЕЙРОННИХ МЕРЕЖ В ЗАВДАННЯХ ІДЕНТИФІКАЦІЇ

В.1 Моделі систем, використовувані при нейромережевій ідентифікації

Основним питанням в нейрорегулювання є отримання адекватних моделей динамічних систем. Питання про існування дискретних моделей, або іншими словами про спостереженість нелінійних систем, є ключовим. У загальному випадку завдання побудови математичної моделі об'єкту полягає у виборі структури і оцінці її параметрів так, щоб при використанні критерію мінімуму деякої функції різниці розрахункових і експериментальних даних дотримувалася умова близькості моделі досліджуваному процесу. Застосування нейронних мереж як моделі може служити альтернативою класичним методам математичного опису систем, оскільки в цьому випадку точне знання внутрішніх процесів не є необхідною умовою моделювання.

При нейронній ідентифікації нелінійних систем найчастіше знаходять застосування наступні основні нелінійні дискретні моделі.

У найбільш загальному вигляді, прийнятому в класичній теорії управління, опис стохастичних нелінійних систем в просторі станів розглядався в роботі Наренди К.С. і Партасараті К [81].

$$z(k+1) = \Psi[z(k),\ u(k),\ \varphi(k)]\ ;$$

$$y(k) = \Phi[z(k),\ u(k),\ \phi(k)],$$

де $z(k) = [z_1(k),\ z_2(k),...,\ z_{k_z}(k)]^{\mathrm{T}}$ – вектор стану системи;

$u(k) = [u_1(k),\ u_2(k),...,\ u_{k_u}(k)]^{\mathrm{T}}$ – вектор вхідних сигналів;

$y(k) = [y_1(k),\ y_2(k),...,\ y_{k_y}(k)]^{\mathrm{T}}$ – вектор вихідних сигналів;

$\Psi[\cdot],\ \Phi[\cdot]$ – деякі статичні нелінійні функції;

$\varphi(k),\ (k)$ – шум процесу і шум вимірювань відповідно.

Леонарітіс І. Дж. і Биллингз С.Д.[82] запропонували для опису нелінійних систем використовувати так звану нелінійну модель авторегресії ковзаючого середнього з додатковими вхідними сигналами (NARMAX-модель).

$$y(k) = f[y(k-1),...,y(k-n_y),\ u(k-1),...,u(k-n_u),\ e(k-1),...,e(k-n_e)] + e(k) \qquad \text{(B.1)}$$

де $y(k) \in \mathbf{R}^{M\times 1}$ – вихідний сигнал об'єкту;

$u(k) \in \mathbf{R}^{N\times 1}$ – вхідний сигнал об'єкту;

$e(k) \in \mathbf{R}^{M\times 1}$ – різниця вихідних сигналів об'єкту і моделі;

$f[\cdot]$ – нелінійна функція перетворення $f:\ \mathbf{R}^{(k_yM+k_uN+k_eM)\times 1} \to \mathbf{R}^{M\times 1}$;

$n_y,\ n_u,\ n_e$ – порядки запізнювання по вихідному, вхідному сигналам об'єкту і помилці вимірювань відповідно.

Скалярна форма рівняння (B.1) має вигляд:

$$y_1(k) = f_1[y_1(k-1),...,y_1(k-n_y),y_2(k-1),...,y_2(k-n_y),y_M(k-1),...,y_M(k-n_y),$$

$$u_1(k-1),...,u_1(k-n_u),u_2(k-1),...,u_2(k-n_u),u_N(k-1),...,u_N(k-n_u),$$

$$e_1(k-1),...,e_1(k-n_e),e_2(k-1),...,e_2(k-n_e),e_M(k-1),...,e_M(k-n_e)] + e_i(k) \qquad \text{(B.2)}$$

Нелінійні системи можуть бути представлені нелінійною NARX-моделлю:

$$y(k) = f[y(k-1),...,y(k-n_y),\ u(k-1),...,u(k-n_u)] + e(k) \qquad \text{(B.3)}$$

або в скалярному вигляді

$$y_1(k) = f_1[y_1(k-1),...,y_1(k-k_y),y_2(k-1),...,y_2(k-k_y),y_M(k-1),...,y_M(k-k_y),$$

$$u_1(k-1),...,u_1(k-k_u),u_2(k-1),...,u_2(k-k_u),u_N(k-1),...,u_N(k-k_u)] + e_i(k). \qquad \text{(B.4)}$$

Представлення об'єктів у вигляді NARMAX або NARX моделей грають фундаментальну роль при дослідженні нелінійних об'єктів за допомогою штучних нейронних мереж. При цьому важливо розглянути узагальнення моделей (B.1) - (B.4), що враховує різні види нелинейностей по вхідних і вихідних змін-

них. Наренда виділив чотири такі моделі:

– модель 1 (рис. В.1):

$$y(k+1) = \sum_{i=1}^{k_y-1} a_i y(k-i) + g[\, u(k),\, u(k-1),\ldots,u(k-n_u+1)]\,; \qquad \text{(В.5)}$$

– модель 2 (рис. В.2):

$$y(k+1) = f[\, y(k),\, y(k-1),\ldots,y(k-n_y+1)] + \sum_{i=1}^{k_u-1} b_i u(k-i)\,; \qquad \text{(В.6)}$$

– модель 3 (рис. В.3):

$$y(k+1) = f[\, y(k),\, y(k-1),\ldots,y(k-n_y+1)] + g[\, u(k),\, u(k-1),\ldots,u(k-n_u+1)]\,; \qquad \text{(В.7)}$$

– модель 4 (рис. В.4):

$$y(k+1) = f[\, y(k),\, y(k-1),\ldots,y(k-n_y+1),\, u(k),\, u(k-1),\ldots,u(k-n_u+1)]\,. \qquad \text{(В.8)}$$

Модель (В.4) є найбільш загальною зі всіх моделей вхід/вихід, описаних вище. При цьому модель (В.8) є аналітично вирішуваною і найбільш зручною для практичних застосувань. Внаслідок цього вона використовується в роботі при ідентифікації нелінійних об'єктів за допомогою штучних нейронних мереж.

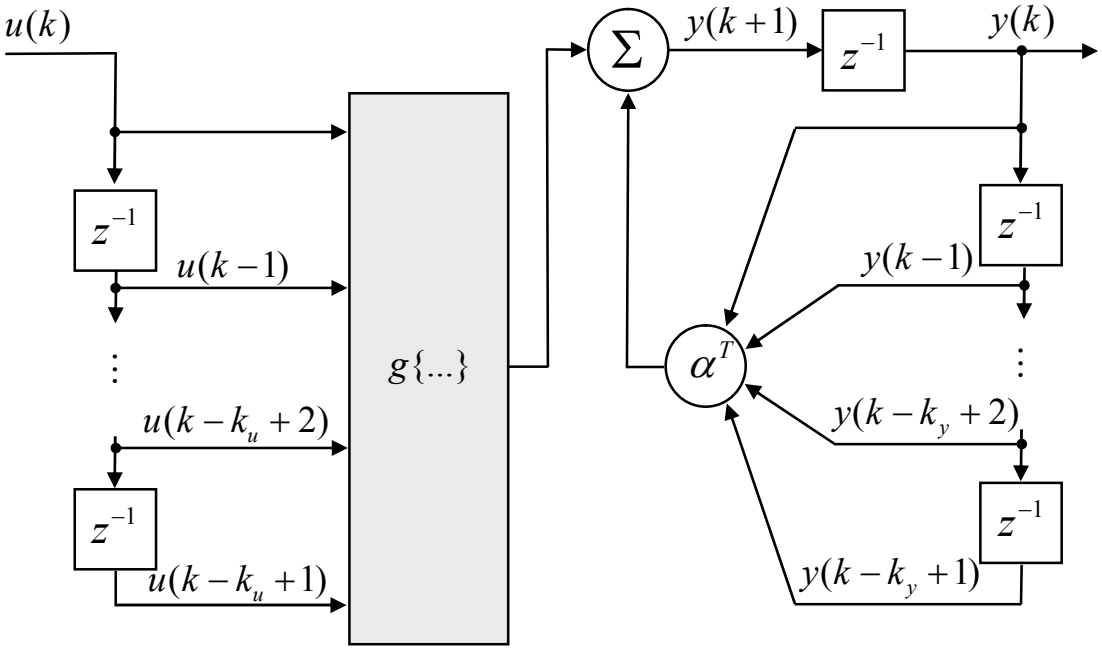


Рисунок В.1 – Модель 1. Вихід моделі нелінійно залежить від вхідних сигналів і лінійно від вихідних

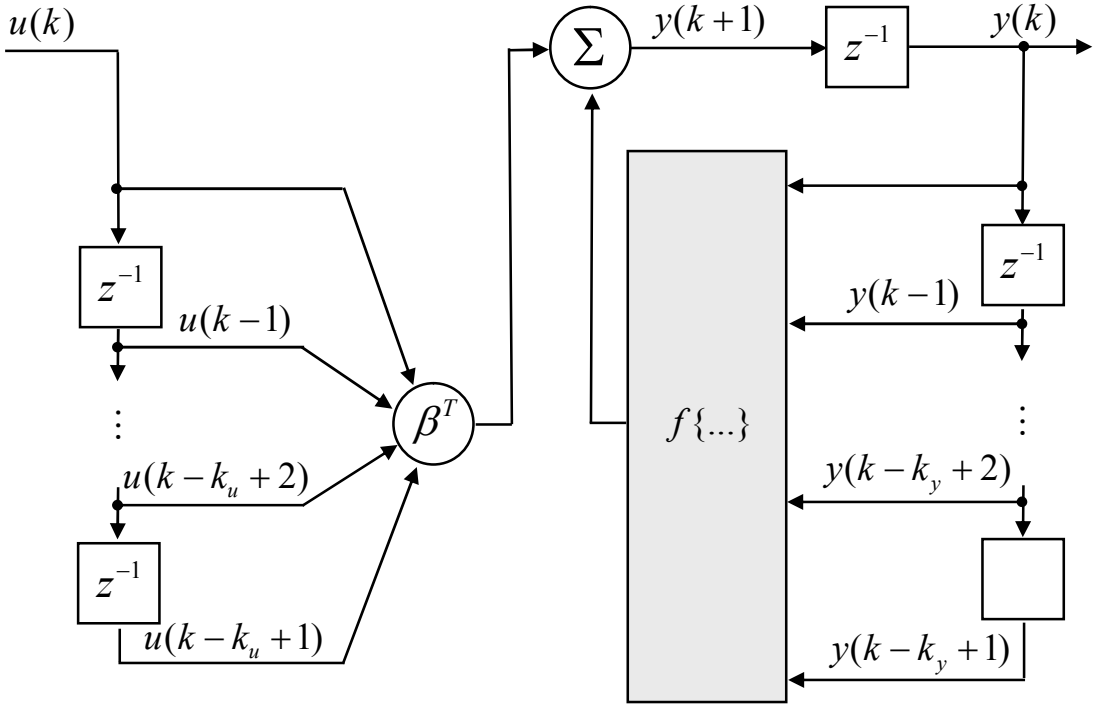


Рисунок В.2 – Модель 2. Вихід моделі лінійно залежить від вхідних сигналів і нелінійно від вихідних

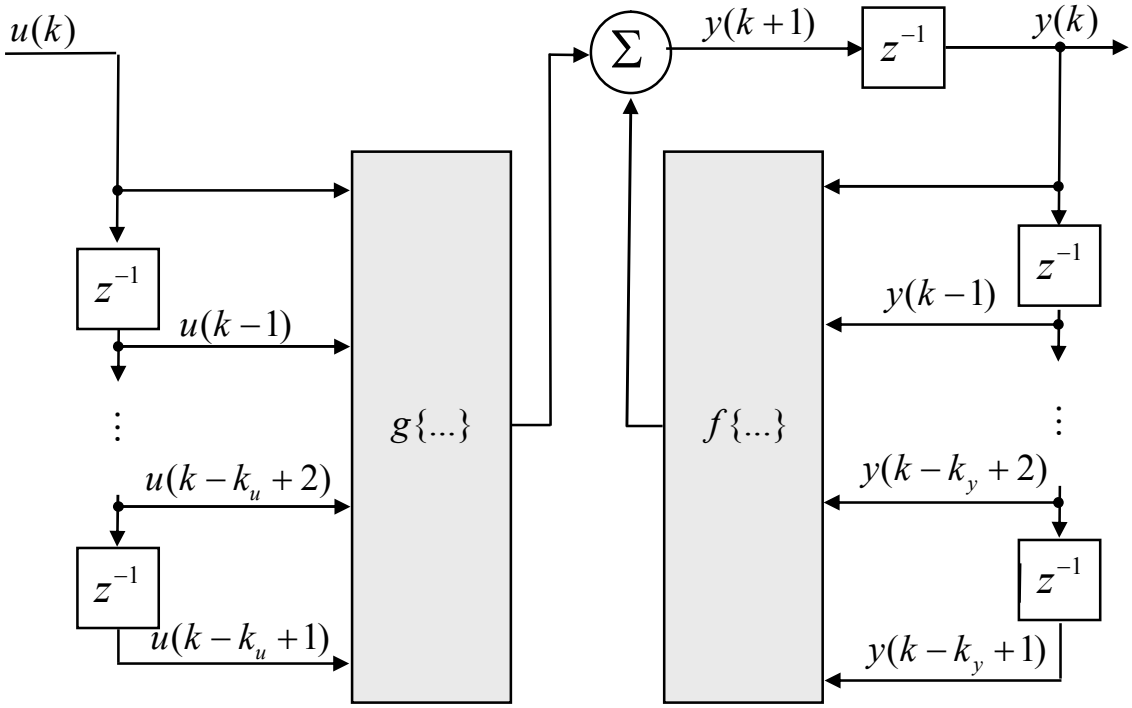


Рисунок В.3 – Модель 3. Результуючий сигнал є адитивною функцією нелінійних перетворень вхідних і вихідних сигналів

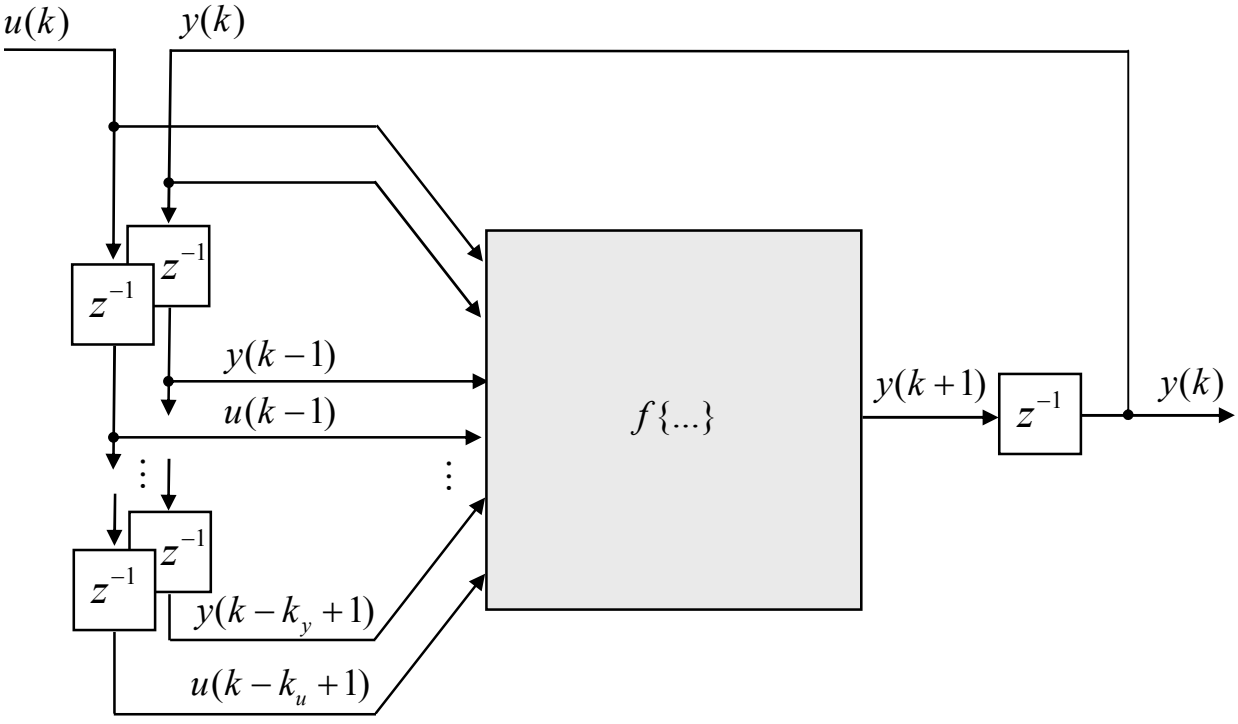


Рисунок В.4 – Модель 4. Вихід моделі нелінійно залежить як від вхідних, так і від вихідних сигналів

В.2 Нейронні мережі як універсальні моделі

Оскільки тип і взаємозв'язок нелінійностей в об'єкті, як правило, не відомі, для моделювання слід використовувати якомога більш універсальні структури моделей. Нейронні мережі є саме такими структурами.

Нелінійні моделі в просторі станів можуть бути реалізовані за допомогою рекурсивних нейронних мереж з внутрішніми зворотними зв'язками (мережі Елмана). Ідентифікація параметрів цих мереж вимагає спеціальної і щодо трудомісткої форми алгоритму навчання. Крім того, доказ стійкості таких нелінійних динамічних моделей практично неможливо зробити .

У роботі використовуються прямонаправленные мережі, що характеризуються наявністю зв'язків між нейронами тільки в прямому напрямі без зворотних зв'язків усередині мережі, – багатошаровий персептрон. Найбільш часто використовувана мережа, багатошаровий персептрон (БП). За допомогою БП можна апроксимувати з бажаною точністю не тільки будь-які статичні функції. Попередні значення вхідних/вихідних координат у вхідному векторі дозволяють додати прямонаправленным мережам динамічні властивості.

Володіючи схожими з радіально-базисними мережами (РБМ) властивостями, БП забезпечує компактніше представлення нелінійного об'єкту, оскільки число параметрів в мережі РБМ експоненціально зростає із збільшенням розмірності вхідного простору, тоді як для БП ця залежність лінійна. Важливою є наочність, що виражається в тому, що персептронне представлення моделі еквівалентне традиційному. Це дозволяє використовувати для визначення параметрів ШНМ добре розвинені методи оцінювання, використовувані при традиційному підході.

Модель системи в просторі станів може бути успішно реалізована за допомогою прямонаправленных мереж, якщо змінні стани сформувати на виході мережі і потім завести їх на вхід мережі із затримкою. Але для цього потрібний, щоб всі змінні стани були вимірювані, оскільки тільки в цьому випадку можна забезпечити тренування мережі.

NARX мережеві моделі параметризуються лінійно і тому використовують стандартні методи тренування мережі. Внаслідок цього саме даний вид моделі найчастіше використовується при нейронній ідентифікації систем. Для простоти викладу надалі розглядаємо системи з одним входом і одним виходом SISO (singl input singl output), хоча всі висновки можуть бути поширені і на MIMO (multi input multi output) системи. Прямій нейронній NARX моделі нелінійної динамічної системи відповідає наступний вираз:

$$\hat{y}(k) = f\left(y(k-1)...y(k-n_y), u(k-n_k)...u(k-n_u-n_k+1), \mathbf{w}\right), \qquad \text{(B.9)}$$

а нейронній інверсній NARX моделі:

$$\hat{u}(k) = g\left(y(k+n_k)...y(k+n_k-n_y), u(k-1)...u(k-n_u+1), \mathbf{w}\right) \qquad \text{(B.10)}$$

де f, g – функції, що реалізовуються нейронними мережами;

u, y – вхідна і вихідна координати відповідно;

$\hat{u}$, $\hat{y}$ – значення координат на виході мережі;

$\mathbf{w}$ – матриця лінійних параметрів мережі;

n_k – запізнювання в системі, найчастіше $n_k = 1$.

Вирази (B.9) і (B.10) можна представити графічно у вигляді структурних схем, показаних на рис. B.5 і рис. B.6. NARMAX моделі характеризуються вищою якістю ідентифікації при зашумленных тренувальних даних. У [83] пропонується спосіб модифікації критерію якості алгоритму тренування. Результати ідентифікації за допомогою NARX моделей наближаються в цьому випадку до результатів, отриманих за допомогою NARMAX моделей. Дані NARX нейромоделі можуть бути успішно застосовані для вирішення поставлених завдань.

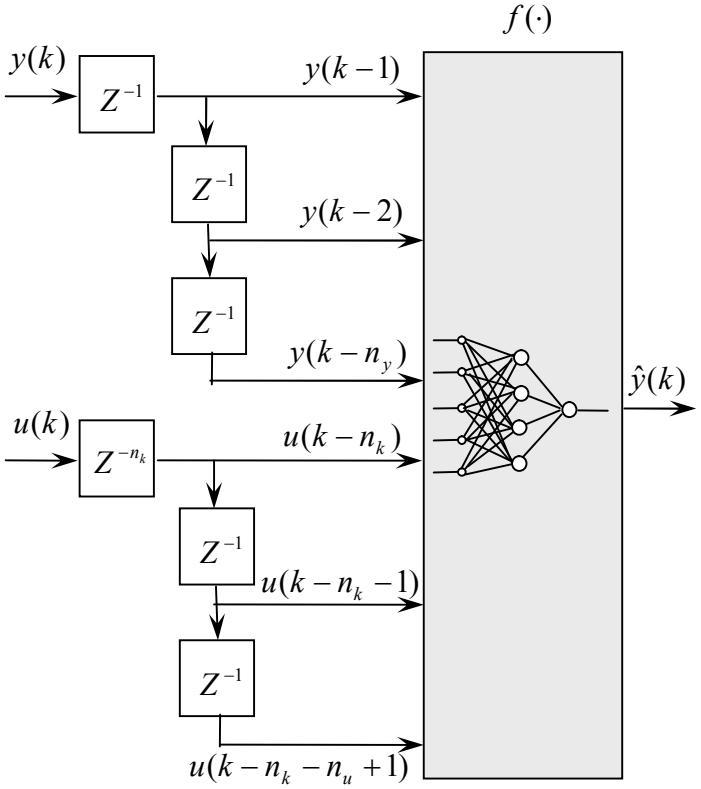


Рисунок В.5– Пряма мережева NARX модель

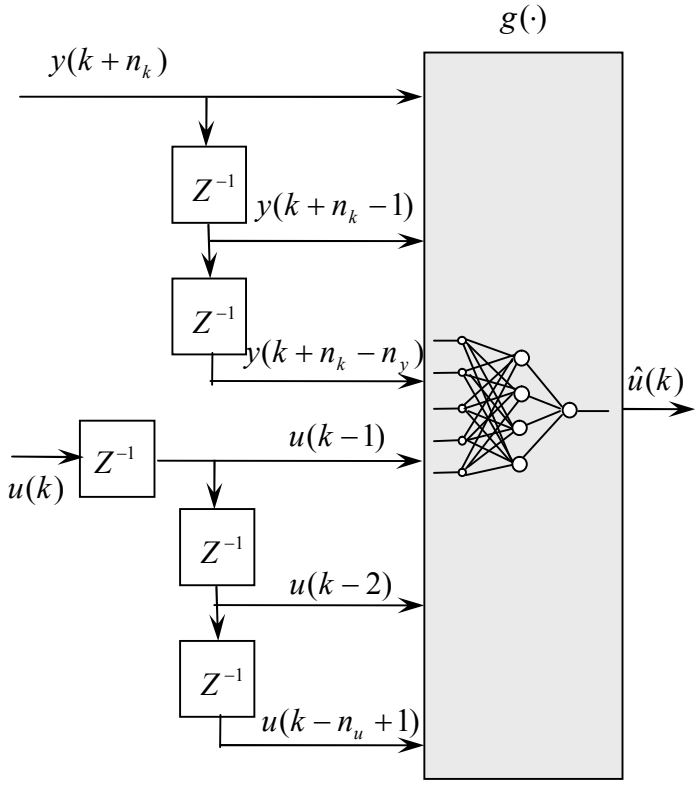


Рисунок В.6 – Інверсна мережева NARX модель

В.3 Ідентифікація динамічних систем за допомогою нейроннихмереж

Завдання теорії регулювання, що займається пошуком моделей динамічних систем на основі апріорної інформації і вимірюваних вхідних/вихідних сигналів системи, носить назву ідентифікація. Відомі класичні методи ідентифікації нелінійних систем застосовні тільки у поєднанні з певним типом моделі (наприклад, ряди Вальтера, моделі Хаммерштейна і Вінера), що описує певний клас нелінійних динамічних систем. У роботі під термінами «ідентифікація системи», «тренування» і «навчання» розуміється процес підстроювання вагових коефіцієнтів мережі за допомогою градієнтних методів. Метою цього процесу є по можливості точне наближення поведінки моделі до спостережуваної поведінки динамічної системи.

Принципово розрізняють пряму і інверсну модель об'єкту. На рис. В.7 представлені схеми тренування цих нейронних моделей. Для спрощення на рис. В.7 не зображені ланки затримки за часом, що формують попередні значення координат на вході нейронних мереж.

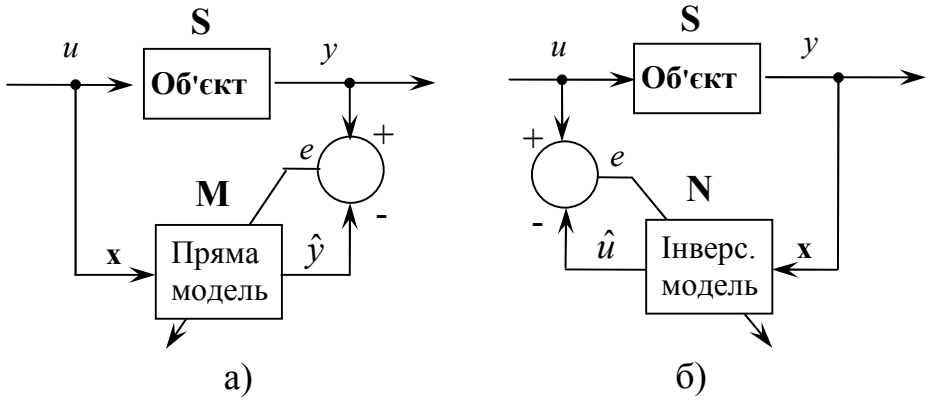


Рисунок В.7 – Схеми тренування прямої (а) і інверсної (б) моделей

Вхідний сигнал x нейронній моделі об'єкту **M** на рис. В.7,а ідентичний вхідному сигналу об'єкту **S**. Для інверсної нейромоделі **N** як вхідний сигнал використовується вихідний сигнал об'єкту **S** тоді як нейромодель повинна відновити вхідний сигнал об'єкту (див. рис. В.7,б). Поки присутня помилка між виходом нейромоделі і її зразковим сигналом, ця помилка використовується для

тренування мережі. У обох випадках ідентифікації обчислюється градієнт функціонала якості (див. таблицю В.1). Функціонал якості найчастіше при цьому квадратичний.

Таблица В.1 – Обчислення градієнта при тренуванні нейромоделей

	Пряма модель	Інверсна модель
Бажаний вихідний сигнал	y	u
Вихідний сигнал нейромоделі	$\hat{y} = \mathbf{M}(\mathbf{x}, \mathbf{w})$	$\hat{u} = \mathbf{N}(\mathbf{x}, \mathbf{w})$
Помилка навчання	$e = y - \hat{y}$	$e = u - \hat{u}$
Градієнт функціонала якості	$\frac{\partial J}{\partial \mathbf{w}} = \frac{\partial J}{\partial e}\frac{\partial e}{\partial \mathbf{w}} = -e\frac{\partial \hat{y}}{\partial \mathbf{w}}$	$\frac{\partial J}{\partial \mathbf{w}} = \frac{\partial J}{\partial e}\frac{\partial e}{\partial \mathbf{w}} = -e\frac{\partial \hat{u}}{\partial \mathbf{w}}$

Оскільки $\frac{\partial \hat{y}}{\partial \mathbf{w}}$ і $\frac{\partial \hat{u}}{\partial \mathbf{w}}$ обчислювані, то це дозволяє вести підстроювання вагових коефіцієнтів мережі згідно розглянутим градієнтним методам. Результатом цього буде мінімізація функціонала якості. Оскільки набір тренувальних даних однозначно відображає лише поведінку модельованого об'єкту $\mathbf{S}$ у робочому діапазоні, то після завершення тренування маємо $\mathbf{M} \approx \mathbf{S}$ і $\mathbf{N} \approx \mathbf{S}^{-1}$.

Типова послідовність дій, що приводить до формування нейромоделі (ідентифікації) динамічної системи, відображена в таблиці В.2.

Таблиця В.2 – Ідентифікація за допомогою нейронних мереж

Шаг	Дія
1	Знімання тренувальних даних
2	Вибір структури мережі
3	Вибір типу моделі
4	Тренування мережі
5	Тестування нейромоделі
6	При незадов. результаті тестування, перейти до п.3 або до п.2

Оскільки тренувальні дані є носієм інформації про систему, що ідентифікується, то наявності, кількості і якості цих даних слід приділити особливу увагу. Тренувальних дані повинні мати наступні характеристики: якісна дискретизація, рівномірність дисперсії і величини варіації, рівномірність амплітуд всіх сигналів і їх середньоквадратичних значень.

При отриманні позитивного результату апроксимації, слід також перевірити здібність нейромоделі до узагальнення, тобто її здатність повторювати істотні характеристики об'єкту. Для цього тесту необхідний ще один набір даних, що не брав участь при тренуванні мережі. Цей тест дає остаточний висновок про адекватність нейромоделі.

Представлені теоретичні положення по нейронній ідентифікації, а також характеристики прямонаправлених мереж є основою успішного регулювання нелінійних динамічних систем.

ДОДАТОК Г
ФАЙЛ ВИХІДНИХ ДАНИХ

```
clear;clc;echo on;

% Система управління електроприводом переміщення моста
% мостового крану вантажопідіймальністю Q=37т
% 2025 рік

% Вихідні дані
kd=0.179;
CF=1/kd;
In=84;
Mn=In/kd;
ktp=59.8;
kn=0.05454;
kc=0.143;
Tmc=0.00317;
Ta=0.071;
Te=0.061;
Re=0.36;
Jd=2.4;
Jm=18.4;
Js=Jd+Jm;
b12=0;
w0=6;
c12=(w0^2)*(Jd*Jm)/(Jd+Jm);
Uz=24;
 Tme=2*Tmc+Ta;
kic=Re/(2*Tmc*kc*ktp);
kpc=kic*Te;
kpe=Js*kd^2*kc/(2*Tme*kn);
kie=kpe/(4*Tme);
 % Матриці з розкритим контуром струму (матр.без скор.)
A=[-b12/Jm  1/Jm  b12/Jm  zeros(1,6);
-c12 0 c12 zeros(1,6);
b12/Jd -1/Jd -b12/Jd CF/Jd zeros(1,5);
0 0 -CF/(Re*Te) -1/Te 1/(Re*Te) 0 0 0 0;
0 0 0 (-kc*kpc*ktp)/Tmc   -1/Tmc   ktp/Tmc (kpc*ktp)/Tmc
(kpe*kpc*ktp)/Tmc  -(kpe*kpc*ktp)/Tmc;
0 0 0 -kic*kc  0 0 kic  kpe*kic -kpe*kic;
zeros(1,7)  kie -kie;
zeros(1,7) -1/Ta 0;
0 0  kn/(kd*Ta) zeros(1,5) -1/Ta ]
B=[0 -1/Jm;
zeros(6,2);
1/Ta 0;
0 0]
C=zeros(9,9);
C(1,1)=1; C(2,2)=1;C(3,3)=1;C(4,4)=1;C(5,5)=1; C(6,6)=1; C(7,7)=1;
C(8,8)=1; C(9,9)=1;
D=zeros(9,2);
```