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 ПІДЙОМУ

(тема кваліфікаційної роботи)

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

_____/__Даніїл ЗІПУНОВ_
(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)_____Даніїл ЗІПУНОВ _____
(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 підйому_____
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 підйому 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 підйому. Схема кінематична. 3. Нейронні мережі. Структурні схеми. 4. Навчання нейронних мереж. Метод зворотного розповсюдження помилки. 5. Системи нейрорегулювання. Структурні схеми. 6. Управління з прогнозом. Математична модель. 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 _____ __Даніїл ЗІПУНОВ__
(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 підйому мостового крана.	09.09.2025-18.09.2025	
2	Розрахунок системи автоматичного регулювання координат електроприводу. Дослідження динамічних характеристик системи.	19.09.2025-15.10.20254	
3	Дослідження системи з урахуванням пружних властивостей підіймального канату. 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) _____ __Даніїл ЗІПУНОВ_
(підпис) (ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка складається з: 224 сторінки, 63 рисунки, 2 таблиці, 99 використаних джерел, 5 додатків.

Метою даної магістерської роботи є створення та обґрунтування сучасних підходів до синтезу системи керування електроприводом підйомного механізму мостового крана, які забезпечують покращення динамічних та якісних показників його роботи шляхом застосування нейромережевих регуляторів.

У процесі дослідження розрахунковими методами, що спираються на базові принципи теорії автоматичного керування, визначено основні параметри системи керування електроприводом. Для опису поведінки багатомасових електромеханічних систем використано методи математичного моделювання, на основі яких побудовано відповідні математичні моделі. Застосування інструментарію штучних нейронних мереж дало змогу виконати синтез нейрорегулятора для керування двомасовою електромеханічною системою та оцінити його ефективність.

За допомогою засобів комп'ютерного моделювання в середовищі MATLAB проведено комплексні дослідження перехідних процесів та динамічних характеристик нейромережевої системи керування, що дозволило оцінити її працездатність у різних режимах роботи.

У ході виконання роботи доведено наукову доцільність та практичну перспективність впровадження нейромережевих підходів для керування багатомасовими системами електропривода підйомних механізмів. Розроблено функціональну структуру системи координатного керування електроприводом, визначено параметри автоматичного регулятора та побудовано детальну математичну модель, що враховує пружні властивості підіймального каната. Проаналізовано динаміку двомасової системи та досліджено поведінку нейромережевого регулятора типу NN Predictive Controller, синтезованого в середовищі MATLAB. Отримані результати підтвердили, що запропонована

структура нейромережевого керування забезпечує високий рівень якості та стійкості функціонування системи.

Основні результати та методичні напрацювання магістерської роботи впроваджені в навчальний процес кафедри АМЕТ ННІ «УІПА» Харківського національного університету імені В.Н. Каразіна та використовуються під час викладання дисциплін, виконання курсових проєктів і проведення практичних занять.

Положення та наукові висновки, отримані в роботі, були представлені та успішно апробовані на науково-практичній конференції студентів ННІ «УІПА» (Харків, 2025 р.).

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

ABSTRACT

An explanatory message consists of: 224 pages, 63 figures, 2 tables, 99 used sources, 5 additions.

The purpose of this master's thesis is to create and substantiate modern approaches to the synthesis of the electric drive control system of the lifting mechanism of a bridge crane, which provide improvement of the dynamic and qualitative indicators of its operation by using neural network controllers.

In the process of research using computational methods based on the basic principles of the theory of automatic control, the main parameters of the electric drive control system were determined. To describe the behavior of multi-mass electromechanical systems, mathematical modeling methods were used, on the basis of which the corresponding mathematical models were built. The use of artificial neural network tools made it possible to synthesize a neuroregulator for controlling a two-mass electromechanical system and assess its effectiveness.

Using computer modeling tools in the MATLAB environment, comprehensive studies of transient processes and dynamic characteristics of the neural network control system were carried out, which allowed assessing its performance in different operating modes.

During the work, the scientific feasibility and practical prospects of implementing neural network approaches for controlling multi-mass systems of electric drives of lifting mechanisms were proven. The functional structure of the coordinate control system of the electric drive was developed, the parameters of the automatic controller were determined, and a detailed mathematical model was built that takes into account the elastic properties of the hoisting rope. The dynamics of the two-mass system were analyzed and the behavior of the neural network controller of the NN Predictive Controller type synthesized in the MATLAB environment was investigated. The results obtained confirmed that the proposed neural network control structure provides a high level of quality and stability of the system's functioning.

The main results and methodological developments of the master's thesis were

implemented in the educational process of the AMET Department of the NNI "UIPA" of the V.N. Karazin Kharkiv National University and are used during teaching disciplines, performing course projects, and conducting practical classes.

The provisions and scientific conclusions obtained in the work were presented and successfully tested at the scientific and practical conference of students of the National Research Institute "UIPA" (Kharkiv, 2025).

Keywords: ARTIFICIAL NERO, NEURAL NETWORKS, BRIDGE CRANE, ELECTRIC DRIVE OF LIFTING MECHANISM, INFERIOR CONTROL SYSTEM, 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/ Сучасний стан промисловості характеризується швидким ускладненням технологічних процесів, зростанням вимог до точності, стабільності та швидкодії обладнання. У міру того як виробничі цикли стають дедалі комплекснішими, зростає і відповідальність систем автоматизації, що мають забезпечувати високу якість продукції у постійно змінних умовах експлуатації. Збільшення обсягів виробництва вимагає від машин не лише більшої швидкості обробки операцій, а й підвищення їх потужності, ресурсності та здатності працювати з широким діапазоном навантажень.

Оскільки саме електропривод є основним руховим елементом більшості сучасних машин, підвищення вимог до технологічного обладнання природно призводить до ускладнення завдань, покладених на системи електроприводів. Вони повинні точно й оперативно виконувати переміщення робочих органів, враховуючи режими розгону, гальмування, реверсу, а також забезпечувати стійке регулювання заданих координат і можливість відпрацювання складних законів зміни керуючих величин.

До механізмів, що найповніше акумулюють ці вимоги, належать вантажопідіймальні крани. У реальних умовах їх робота характеризується непостійним навантаженням, зміною маси, геометричних параметрів і типу вантажів, складністю просторових умов пересування та необхідністю безпечної роботи як у виробничих приміщеннях, так і на відкритих майданчиках. Значні коливання навантаження на електроприводи кранів висувають жорсткі вимоги до точності вибору параметрів електродвигунів, систем автоматичного регулювання та захисту.

Сучасні тенденції у розвитку автоматизованих систем вимагають від систем керування електроприводами високої адаптивності, здатності працювати в умовах невизначеності та змін зовнішнього середовища. Ручне керування в таких умовах стає ненадійним і недостатньо точним, що зумовлює необхідність

переходу до інтелектуалізованих систем управління, здатних аналізувати стан об’єкта в реальному часі та приймати адекватні рішення.

Швидке поширення мікропроцесорної техніки, цифрових систем керування та інтелектуальних алгоритмів відкрило можливість застосування принципово нових стратегій управління. Одним із найперспективніших напрямів є використання адаптивних методів разом із технологіями штучних нейронних мереж (ШНМ), які вже зарекомендували себе в багатьох галузях науки й техніки як ефективний інструмент для роботи з нелінійними, нестаціонарними та важкоформалізованими процесами.

Штучні нейронні мережі, що виникли на межі прикладної математики, комп’ютерних наук, біології та інженерії, перетворилися на самостійний науково-практичний напрям, який об’єднує теорію навчання, методи аналізу й прогнозування, обробку сигналів, оптимізацію та алгоритмічний синтез [1-12]. Сучасні ШНМ здатні не лише запам’ятовувати дані чи формувати асоціації, а й адаптовуватися під час роботи, реконструювати частково втрачено інформацію, демонструвати стійкість до збурень і реалізовувати складні закони керування без необхідності точного математичного опису об’єкта.

Ці властивості роблять нейромережеві технології надзвичайно перспективними для систем керування електроприводами вантажопідйомних механізмів, особливо у випадках, коли врахування пружних властивостей канату та взаємодії мас є критично важливим для забезпечення безпечного та ефективного функціонування. Саме тому обрана тема дослідження є актуальною і відповідає сучасним вимогам розвитку промислової автоматизації.

Мета і завдання дослідження. Головною метою роботи є створення методів синтезу системи керування електроприводом підйомного механізму мостового крана із застосуванням нейромережевих регуляторів, що дозволяють суттєво підвищити якість функціонування системи у динамічних режимах.

Для досягнення поставленої мети передбачено розв’язання таких наукових і прикладних завдань:

1. Провести аналіз існуючих систем керування електроприводами та оцінити можливості інтеграції нейромережевих підходів у ці системи.

2. Розвинути математичний опис динаміки об’єкта управління з урахуванням його реальних фізичних властивостей.

3. Сформувати структурну схему системи керування, що містить нейромережевий регулятор.

4. Синтезувати нейромережеву систему прогнозуючого керування з високими якісними показниками.

5. Виконати комп’ютерне моделювання розробленої системи та провести аналіз отриманих результатів.

Об’єкт дослідження – динамічні процеси у системі керування електроприводом підйомного механізму мостового крана з урахуванням пружності підіймального каната.

Предмет дослідження – моделі та методи синтезу нейромережевих систем керування для багатомасових електромеханічних систем підйомних механізмів.

Методи дослідження. Теоретичною основою роботи є положення класичної та сучасної теорії автоматичного керування. Математичне моделювання застосовано для побудови моделей механізму та аналізу його поведінки. Методи штучних нейронних мереж використані для створення нейромережевих моделей і вибору алгоритмів їх навчання. Принципи нейроуправління лягли в основу синтезу інтелектуальної системи керування. Імітаційне комп’ютерне моделювання виступило інструментом оцінки ефективності запропонованих рішень.

Наукова новизна. Наукову новизну роботи становлять такі результати:

1. Уперше науково доведено ефективність застосування нейромережевих підходів для керування багатомасовою електромеханічною системою підйомного механізму мостового крана.

2. Синтезовано принципово нову систему нейромережевого керування електроприводом із урахуванням пружних властивостей підіймального канату, що демонструє високі якісні показники.

3. Запропоновано математичну модель системи у формі багатошарового персептрона, адаптованого для керування підйомним механізмом.

4. Розвинуто математичні моделі двомасових електромеханічних систем для аналізу складних кінематичних з'єднань.

5. Поглиблено методи синтезу прогнозуючих нейромережевих систем керування для динамічних електромеханічних об'єктів.

Практичне значення. Практична цінність отриманих результатів полягає у створенні нейромережевої системи керування двомасовою електромеханічною системою підйомного механізму, яка забезпечує точне, стабільне та надійне регулювання у реальних експлуатаційних умовах.

Особистий внесок здобувача. Усі наукові результати, винесені на захист, отримані автором особисто. Здобувачем виконано: синтез системи регулювання координат електропривода; розроблення моделі механізму з урахуванням пружних елементів; побудову структурної схеми нейромережевої системи; створення та навчання регулятора NN Predictive Controller у MATLAB; моделювання роботи системи та аналіз її характеристик.

Апробація результатів. Основні результати дослідження були представлені й обговорені на студентській науковій конференції ННІ «УІПА» ХНУ імені В.Н. Каразіна (Харків, 2025 р.).

Структура і обсяг роботи. Магістерська робота містить вступ, чотири розділи, висновки, додатки та список літератури. Загальний обсяг становить 224 сторінок, включаючи 63 рисунки (з них 45 у тексті та 28 на 26 окремих сторінках), 2 таблиці, 5 додатків обсягом 48 сторінок та список із 99 джерел на 9 сторінках.

РОЗДІЛ 1
АНАЛІЗ СТРУКТУР СИСТЕМ НЕЙРОУПРАВЛІННЯ. ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Розвиток теорії нейронних мереж

1.1.1 Історичні передумови та формування підходу

Упродовж останніх десятиліть нейробіологи, фізіологи та нейроанатоми досягли помітного прогресу у вивченні структури та функціональної організації головного мозку людини. Завдяки глибокому аналізу нейронних зв'язків, механізмів передачі сигналів і принципів побудови нервової тканини поступово формується уявлення про складну "електричну архітектуру" мозку. Попри це, повне розуміння інтегральних механізмів його роботи все ще залишається недосяжним через колосальну складність: мозок містить сотні мільярдів нейронів, кожен з яких формує численні синаптичні зв'язки, створюючи багаторівневу динамічну мережу, що перевершує за складністю навіть найпотужніші сучасні суперкомп'ютери.

Поступове накопичення експериментальних даних сприяло створенню математичних моделей, здатних відтворювати окремі аспекти роботи біологічних нейронів. Такі моделі дали змогу проводити дослідження за допомогою цифрових ЕОМ і значно розширили можливості аналізу. Згодом виявилось, що ці абстрактні моделі не лише віддзеркалюють властивості мозку, а й можуть ефективно виконувати низку корисних завдань прикладного характеру.

У результаті сформувалися дві ключові паралельні цілі нейромережевих досліджень: – поглиблене розуміння функціонування мозку на фізіологічному та когнітивному рівнях;

– створення штучних нейронних мереж, здатних відтворювати або навіть розширювати принципи нейронної обробки інформації.

Психологічні дослідження також відіграли значну роль у становленні нейромережевих концепцій. Зокрема, теорія асоціативного навчання Д. Хебба,

запропонована у 1949 році, стала фундаментом для побудови перших алгоритмів адаптації вагових коефіцієнтів у ШНМ. Хоча з того часу підхід Хебба був суттєво розширений і доповнений, його основна ідея – “нейрони, що активуються разом, з’єднуються сильніше” – залишається важливим концептуальним положенням сучасних систем машинного навчання.

У 1950–1960-х роках завдяки об’єднанню знань із біології, медицини та технічних наук були створені перші штучні нейронні мережі. Первісно вони реалізовувалися у вигляді електронних апаратних схем, а згодом їх перенесли в середовище програмного моделювання. Значний внесок у цей період зробили такі дослідники, як М. Мінський, Ф. Розенблатт, Б. Уїдроу та ін. [13–21]. Саме вони розробили перші одношарові нейронні структури – персептрони, що демонстрували здатність розв’язувати низку практичних завдань: прогнозування метеоданих, аналіз медичних сигналів, задачі технічного зору тощо. На той час здавалося, що створення інтелектуальної системи, подібної до мозку, є лише питанням масштабування кількості нейронів.

Однак подальші дослідження виявили фундаментальні обмеження таких мереж. Зокрема, Мінський і Паперт [21] математично довели, що одношаровий персептрон не здатний реалізовувати навіть деякі прості логічні функції, як-от «Виключне АБО». Ці викриті недоліки призвели до спаду інтересу до ШНМ, а багато науковців змінили напрям своїх досліджень.

Ситуація змінилася лише наприкінці 1970-х – на початку 1980-х років, коли інтерес до нейронних мереж знову зріс завдяки прогресу в мікроелектроніці й комп’ютерній техніці. У 1982–1985 рр. Дж. Хопфілд [22, 23] запропонував новий клас мереж – оптимізаційні нейронні мережі з асоціативною пам’яттю. У 1985 році з’явилися перші комерційні нейрокомп’ютери, зокрема Mark III фірми TRW (США). Наприкінці 1980-х нейромережеві технології отримали грантову підтримку в США, Японії та Європі (програми «NITAP Frontiers», «Basic Research in Adaptive Intelligence and Neurocomputing»). До кінця 1980-х років розробки в сфері ШНМ активно впроваджувалися провідними

компаніями, а DARPA започаткувало програму створення високошвидкісних нейрокомп’ютерів для спеціалізованих задач оборонного призначення.

На початку 1990-х років галузь штучних нейронних мереж увійшла в етап інтенсивного розвитку. Кількість міжнародних конференцій, присвячених нейромережевим дослідженням, досягла сотні вже у 1993 році, а до 1994 року обсяг комерційного ринку ШНМ і нейрокомп’ютерів перевищив 2 млрд доларів, демонструючи щорічний приріст близько 50 %.

У 1990-х роках важливе місце зайняв розвиток багатошарових нейронних мереж, заснованих на алгоритмі зворотного поширення похибки (backpropagation). Роботи Румельхарта, Гінтона та Вільямса [24] відкрили шлях до ефективного навчання глибших моделей, що стало фундаментом для сучасних систем машинного навчання.

У цей же період з’явилися нові архітектури: – радіально-базисні мережі (RBF),

– самоорганізувальні карти Кохонена (SOM),

– рекурентні мережі (RNN),

– мережі Хопфілда [25, 26].

З початку 2000-х років почалася ера глибокого навчання (Deep Learning). Роботи Лекуна, Бенджіо та Хінтона [27] визначили базові підходи до побудови глибоких згорткових мереж (CNN), а також модифікованих рекурентних архітектур (LSTM, GRU), здатних обробляти часові послідовності.

У 2010–2020-х роках увага наукової спільноти була зосереджена на питаннях інтерпретованості, стабільності, здатності до узагальнення та енергоефективності моделей. Почали активно розвиватися нейромережеві засоби керування, системи змішаного інтелекту (нейронно-символьні підходи) та методи глибокого підкріплювального навчання (Deep Reinforcement Learning) [28, 29].

Сучасний етап розвитку ШНМ характеризується також появою нанотехнологічних та біоінспірованих підходів, що сприяють створенню принципово нових архітектур нейрокомп’ютерів, здатних працювати з набагато вищою пропускною здатністю і мінімальним енергоспоживанням.

Для ґрунтовного вивчення штучних нейронних мереж необхідні знання з широкого спектра дисциплін: нейрофізіології, когнітивних наук, психології, фізики, теорії управління, теорії обчислень, штучного інтелекту, статистики, розпізнавання образів, комп'ютерного зору, паралельних обчислень тощо. Водночас нейронні мережі, своєю чергою, інтенсифікують розвиток цих галузей, формуючи міждисциплінарний науковий простір.

Значна кількість прикладних досліджень доводить перспективність впровадження нейромережевих систем керування. У літературі наведено багато прикладів їхнього успішного застосування для керування різними технічними об'єктами: літаком [30–31], вертольотом [32], гірничозбагачувальним комплексом [33], автономним автомобілем [34], гібридним двигуном [35], пневмоциліндром [36], спеціальними технічними об'єктами [37], перевернутим маятником [38], електропіччю [39], турбогенератором [40], зварювальним обладнанням [41], системами теплового комфорту та енергозбереження [42], системами кондиціонування повітря [43], електроприводами [44], роботизованими маніпуляторами [45–49]. Ці результати підтверджують ефективність нейромереж у керуванні складними нелінійними динамічними системами.

Основні поняття теорії ШНМ, необхідні для розуміння подальших досліджень у магістерській роботі, наведені в додатках А–Г. Оскільки у даній роботі синтез нейромережевої системи керування виконується на основі багатошарової прямонаправленої мережі (багатошарового персептрона), у додатку А розглянуто модель штучного нейрона, а в додатку Б – структуру багатошарових мереж.

Огляд методів навчання та обґрунтування вибору оптимального алгоритму для реалізації поставлених задач наведені в додатку В.

У додатку Г подано реалізацію узагальненого алгоритму прогнозного керування, який базується на використанні прямонаправленої багатошарової нейронної мережі як нелінійної моделі об'єкта. Цей алгоритм є основою побудови нейрорегулятора NN Predictive Controller, що застосовується в магістерській роботі.

1.2 Ідентифікація динамічних об’єктів з застосуванням нейронних мереж

Важливим етапом розв’язання задач управління, прогнозування стану та моделювання нелінійних динамічних систем є формування їх адекватних математичних моделей. Такі моделі, як правило, будуються на основі теоретичного аналізу та експериментальних досліджень властивостей об’єкта. Теоретичний аналіз, що спирається на вивчення фізичних, хімічних чи інших процесів у системі, дозволяє розробити її математичне представлення, зокрема у вигляді диференціальних рівнянь. Під час експериментального аналізу, використовуючи спостереження за вхідними та вихідними сигналами, отримують параметричні або непараметричні моделі.

Найбільш поширеними є саме параметричні моделі, оскільки вони базуються на виборі структури та визначенні відносно невеликого набору параметрів. Проте, попри значну кількість праць у цій галузі, різноманітні типи нелінійностей ускладнюють побудову універсальної теорії ідентифікації нелінійних систем. Традиційно застосовують методи апроксимації нелінійних залежностей, такі як ряди Вольтерра, моделі Гаммерштейна, Вінера, поліноми Колмогорова–Габора тощо, однак їх практична придатність часто обмежена. Додатковою проблемою є наявність завад у реальних сигналах, що потребує їхньої попередньої фільтрації.

Розглянемо задачу ідентифікації нелінійного динамічного об’єкта, який описується NARMAX-моделлю:

$$\tilde{y}(k) = f[y(k-1),...,y(k-m),u(k-1),...,u(k-n),k] + \xi(k), \qquad (1.1)$$

де $\tilde{y}(i), u(i)$ – значення вихідного та вхідного сигналів у момент часу i;

m, n – порядки запізнювання по входу та виходу;

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

ξ – завада вимірювання вихідного сигналу.

Введемо узагальнений вектор входу розмірності $(n+m)\times 1$:

$$x(k) = [\tilde{y}(k-1), \tilde{y}(k-2), ..., \tilde{y}(k-m), u(k-1), u(k-2), ..., u(k-n)], \quad (1.2)$$

після чого рівняння (1.1) можна переписати у компактному вигляді::

$$\tilde{y}(k) = f[x(k), k] + \xi(k). \quad (1.3)$$

Отже, задача ідентифікації полягає у відтворенні функції $f(\cdot)$ на основі спостережуваних послідовностей $u(k)$ та $\tilde{y}(k)$.

Схема ідентифікації нелінійного об'єкту (1.1) на підставі ШНМ приведена на рис. 1.1.

Схему нейромережевої ідентифікації об’єкта, що описується моделлю (1.1), наведено на рисунку 1.1.

Найпоширенішими структурами для реалізації ідентифікації нелінійних динамічних систем є багатошаровий персептрон (БП) (див. рис. 1.2) та радіально-базисні мережі (РБМ) (див. рис. 1.3), що виконують апроксимацію оператора $f(\cdot)$.

Рівняння багатошарового персептрона має вигляд:

$$\hat{y}(k) = f^q\left[(\omega^q)^T f^{q-1}\left[(\omega^{q-1})^T f^{q-2}\left[...f^1\left[(\omega^1)^T\right]x(k)\right]\right]\right], \quad (1.4)$$

де q – кількість нейронів шару;

ω^i – вагові коефіцієнти *i*-го шаруі;

$f(\cdot)$ – активаційні функції *i*-го шару.

У випадку РБМ модель описується виразом:

$$\hat{y}(k) = \sum_{i=1}^{N} c_i \Phi_i(x), \quad (1.5)$$

де c_i – вагові коефіцієнти, які підлягають визначенню;

$\Phi_i(x)$ - базисні функції.

Як базисні функції $\Phi_i(x)$ часто використовують гаусівські радіальні функції:

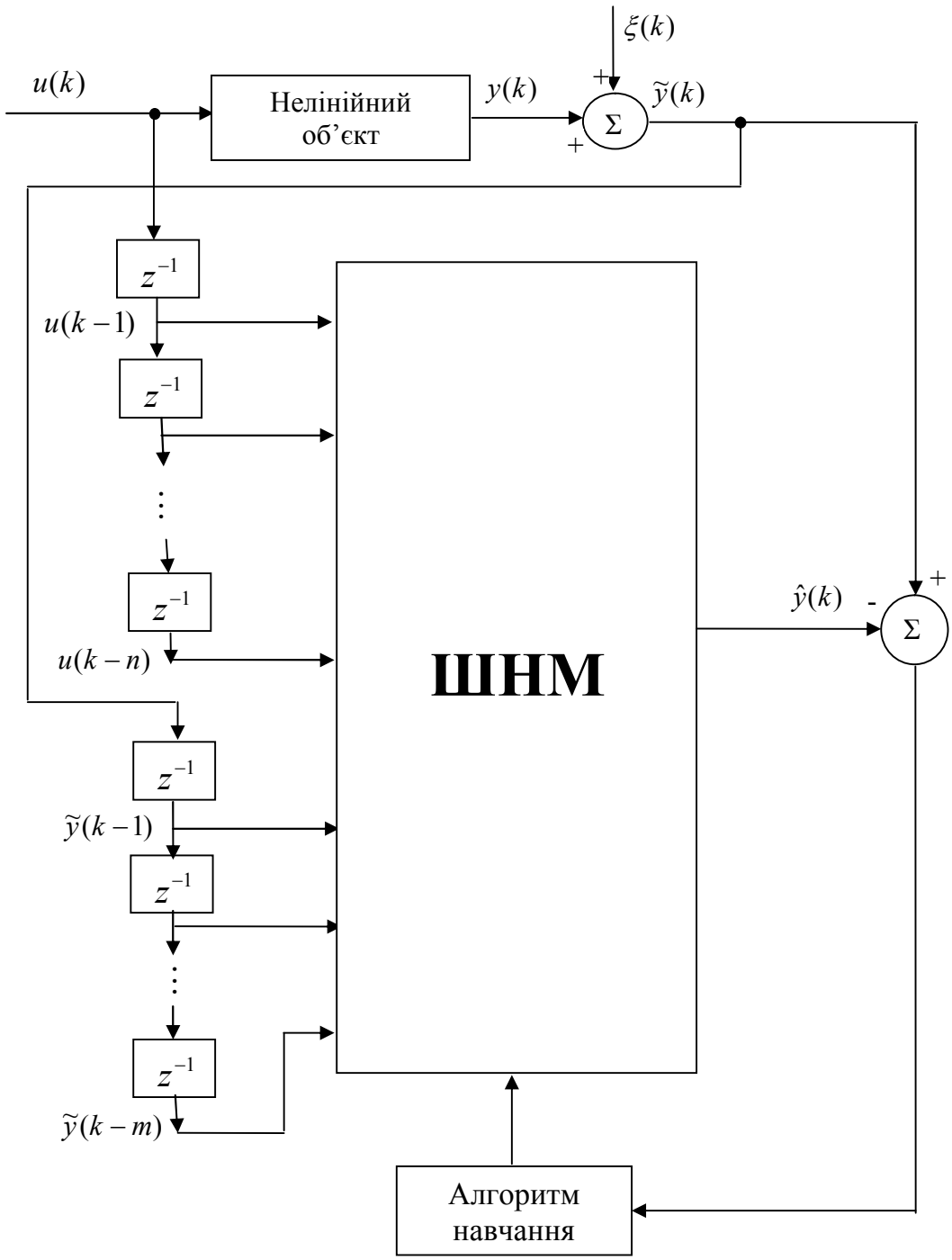


Рисунок 1.1 – Узагальнена схема ідентифікації динамічного об'єкта за допомогою нейронної мережі

$$\Phi_i(x) = \exp\left\{-\frac{\|x-\mu_i\|^2}{\sigma_i^2}\right\}, \qquad (1.6)$$

де μ_i, σ_i – центри і радіуси (ширини) базисних функцій;

$\|\cdot\|$ – евклідовая норма.

Усі невідомі параметри мережі утворюють вектор (зазвичай приймають $\Phi_0(x)=1$).

$$\omega(k)=(c_0(k),c_1(k),\mu_1^T(k),\sigma_1(k),...,c_N(k),\mu_N^T(k),\sigma_N(k))^T,$$

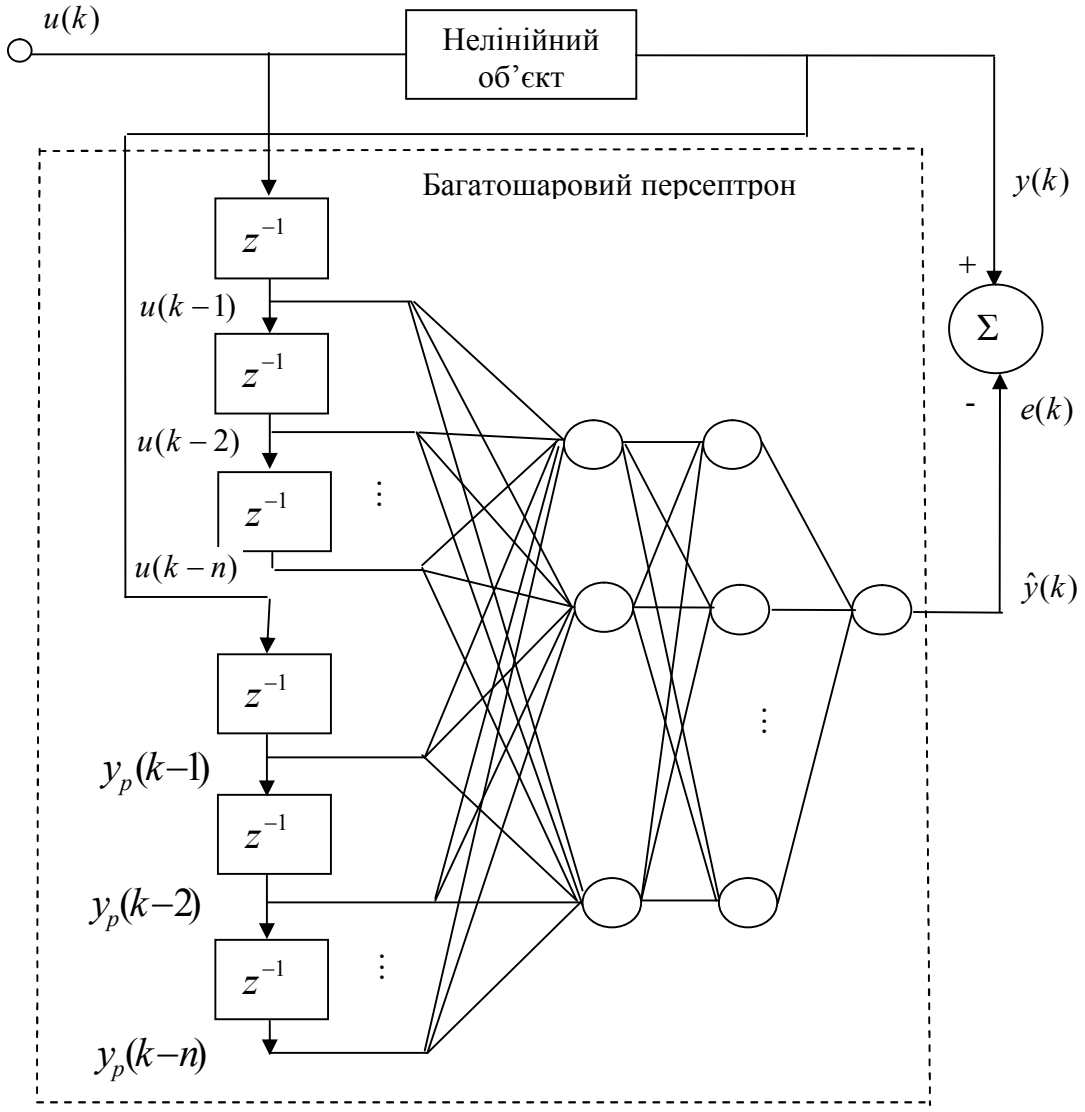


Рис 1.2 – Ідентифікація динамічного об'єкта на основі багатошарового персептрона

Навчання нейронної мережі зводиться до мінімізації функціоналу похибки:

$$J=e^2(k)=M\left\{\left[\tilde{y}(k)-\hat{y}(k)\right]^2\right\}. \qquad (1.7)$$

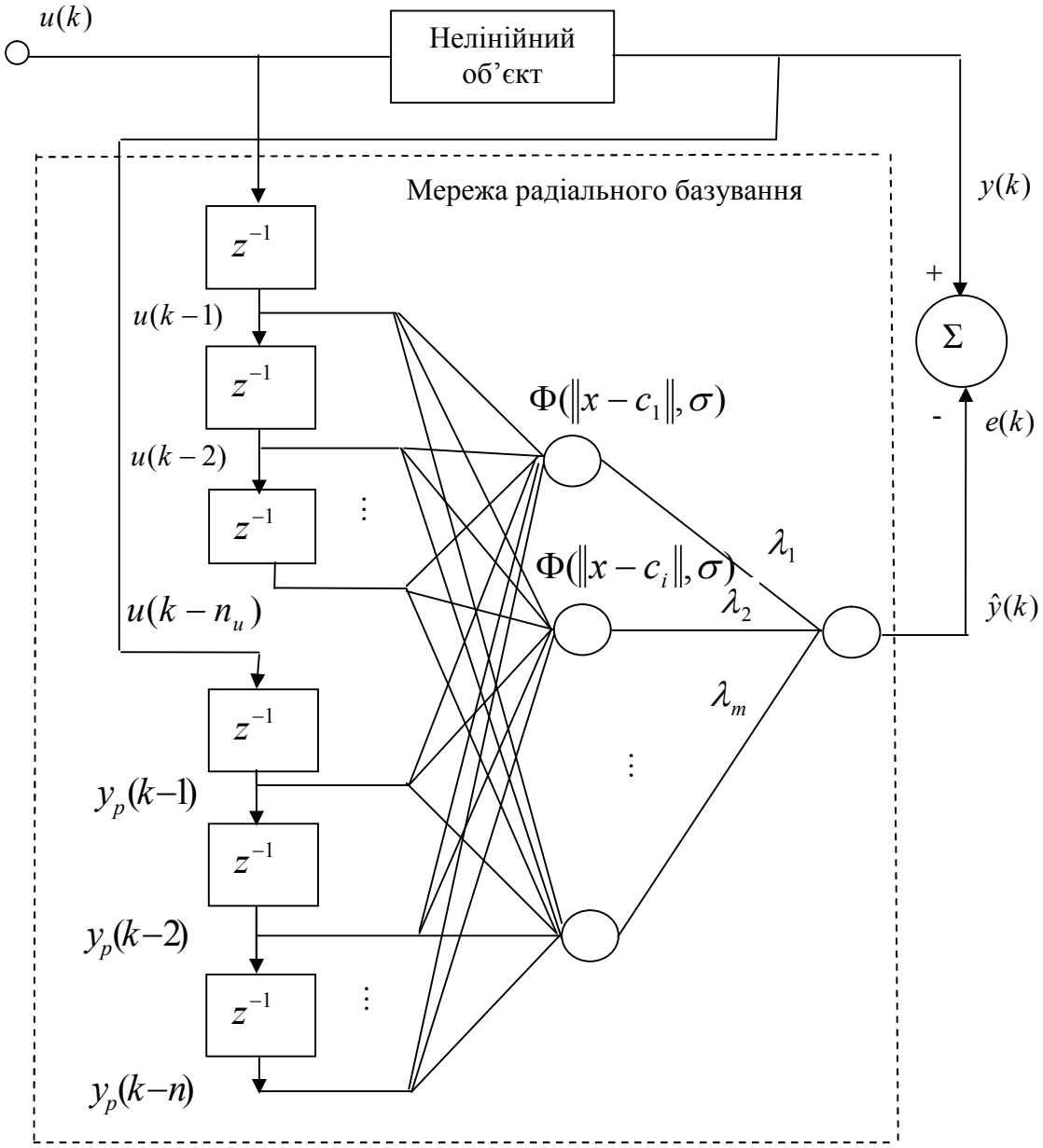


Рисунок 1.3 – Ідентифікація динамічного об'єкта з використанням радіально-базисної мережі

1.3 Нейрокерування динамічними об'єктами

Проектування систем керування реальними об'єктами зазвичай ґрунтується на використанні їхніх математичних моделей. Вибір типу моделі визначає складність майбутнього регулятора. У низці практичних задач застосовують лінеаризовані моделі, які дозволяють одержати параметри регулятора з прийнятною якістю в певній робочій області. Однак для забезпечення оптимальної ро-

боти системи в різних режимах необхідно змінювати параметри регулятора відповідно до нових умов.

Тому актуальним є створення систем керування, що поєднують адаптивні методи з нейромережевими технологіями. Це дає змогу формувати відносно прості моделі на основі ШНМ, а також адаптувати їх параметри разом із параметрами регулятора в процесі роботи системи.

Структурну схему нейромережевої системи адаптивного керування показано на рис. 1.4.

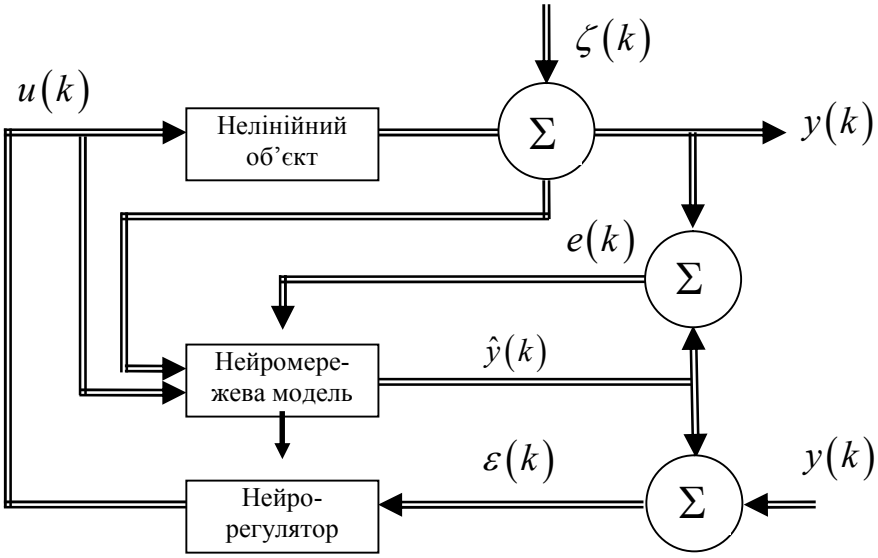


Рисунок 1.4 – Узагальнена структура нейромережевої системи адаптивного керування

Найчастіше для таких задач використовують багатошарові персептрони та радіально-базисні мережі.

Розглянемо багатовимірний нелінійний динамічний об'єкт:

$$y(k) = f\left[y(k-1),...,y(k-m),u(k-1),...,u(k-n)\right] + \xi(k), \qquad (1.8)$$

де $y = \left(y_1(i),...,y_{Ny}(i)\right)^{\mathrm{T}}$; $u(i) = \left(u_1(i),...,u_{Nu}(i)\right)^{\mathrm{T}}$ – вектори вихідних і керуючих впливів розмінностей $N_y \times 1$ і $N_u \times 1$;

$f(\cdot) = \left(f_1(\cdot), f_2(\cdot),..., f_{Ny}(\cdot)\right)^{\mathrm{T}}$ – невідома нелінійна векторна функція;

$\zeta(k)=\left[\zeta_1(k),...,\zeta_{Ny}(k)\right]^{\mathrm{T}}$ – вектор завад.

Введемо вектор узагальнених сигналів:

$$x(k)=\left[y^{\mathrm{T}}(k-1),...,y^{\mathrm{T}}(k-m),u^{\mathrm{T}}(k-1),...,u^{\mathrm{T}}(k-n)\right]^{\mathrm{T}}, \qquad (1.9)$$

тоді (1.8) можна записати як

$$y(k)=f\left(x(k)+\zeta(k)\right). \qquad (1.10)$$

Навчання мережі здійснюється за тим самим принципом, що й під час ідентифікації: параметри налаштовуються таким чином, щоб мінімізувати квадратичний критерій (1.7), порівнюючи сигнали об'єкта $y(k)$ з виходами моделі $\hat{y}(k)$.

Після завершення етапу навчання мережа виконує функцію регулятора. Саме керування полягає у визначенні керуючих дій $u(k)$, що мінімізують критерій:

$$J(k)=\left\|\varepsilon(k)\right\|^2=\frac{1}{2}\left\|y^*(k)-\hat{y}(k)\right\|^2, \qquad (1.11)$$

де $y^*(k)$ – бажані значення вихідних сигналів.

Для пошуку мінімуму функціоналу (1.11) можуть застосовуватися різноманітні рекурентні оптимізаційні алгоритми, включаючи градієнтні:

$$u(k)=u(k-1)+\gamma(k)\left(\nabla_u\varepsilon(k)\right), \qquad (1.12)$$

де $\nabla_u\varepsilon(k)=\dfrac{\partial\varepsilon(k)}{\partial u(k)};\ \gamma(k)>0.$

1.4 Аналіз архітектур нейромережевих систем управління. Вибір оптимальної структури

Історично розвиток систем управління (СУ) відбувався поступово, через низку логічно послідовних етапів. До ключових етапів можна віднести:

- розробку концептуальної моделі СУ;
- створення моделі СУ відповідно до обраної концепції;
- аналіз отриманих результатів і їх порівняння з очікуваними характеристиками;
- вдосконалення та модернізацію початкової концепції на основі аналізу та експериментальних даних.

На кожному з цих етапів проводяться теоретичні дослідження, які допомагають визначити напрямки вдосконалення концепції та розширити її застосування на суміжні сфери.

Схожа логіка спостерігається у розвитку систем управління, що базуються на штучних нейронних мережах (ШНМ). Водночас слід зауважити, що апаратні та програмні реалізації нейромережевих СУ часто випереджають сучасне теоретичне розуміння процесів, що в них відбуваються, і існуючих проблем.

Сьогодні існує велика кількість різних структур систем нейрорегулювання. Деякі з них комбінують аналітичні закони регулювання з нейромоделями: нейронна мережа при цьому відновлює елементи матриць моделі об'єкта. Такі підходи належать до структурованих методів управління. Натомість неструктуровані методи регулювання дозволяють повністю розкрити потенціал апроксимації нейронних мереж. Далі стисло розглянуто сучасні підходи до побудови нейромережевих СУ.

Однією з класичних моделей є система з зворотним зв’язком із коефіцієнтами, що налаштовуються у реальному часі, наприклад, самоналагоджувальний регулятор Астрома [50]. Коефіцієнти такого контролера змінюються протягом кожного циклу управління відповідно до оцінки параметрів об’єкта. Блок-схема такої системи представлена на рис. 1.5.

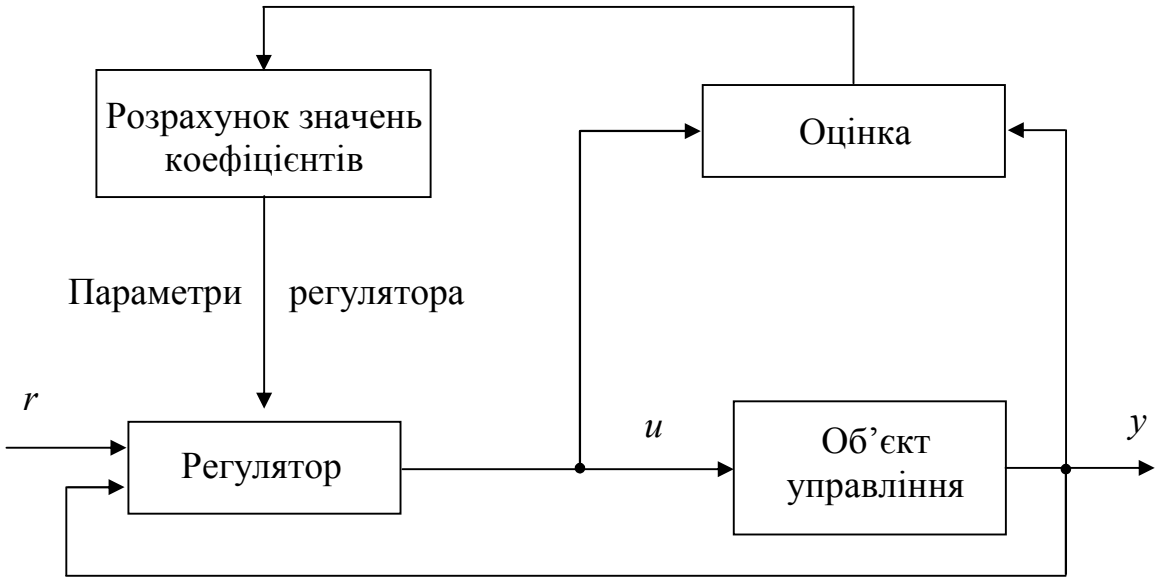


Рисунок 1.5 – Блок-схема управління із зворотним зв’язком і динамічно регульованими коефіцієнтами

Ще одним відомим підходом є управління за моделлю Ляпунова. Системи адаптивного типу, які використовують еталонну модель Ляпунова, проектуються так, щоб вихідний сигнал керованої системи у підсумку збігався з вихідним сигналом заздалегідь визначеної моделі з бажаними динамічними властивостями [51]. Така система гарантує асимптотичну стійкість, тобто керована система відстежує еталонну модель із нульовою похибкою, а перехідні процеси у фазі адаптивного або навчального управління мають обмежені, передбачувані межі. Блок-схема адаптивного управління з еталонною моделлю зображена на рис. 1.6.

Системи управління на основі ШНМ виступають альтернативою класичним методам. Їх ефективність пояснюється тим, що мережа з двома шарами і достатньою кількістю вузлів у прихованому шарі може апроксимувати будь-яку функцію дійсних чисел з необхідною точністю, що випливає з теореми Вейєрштраса [52]. Завдяки цьому навіть нейронні мережі з одним прихованим шаром можуть успішно застосовуватись для задач ідентифікації та управління.

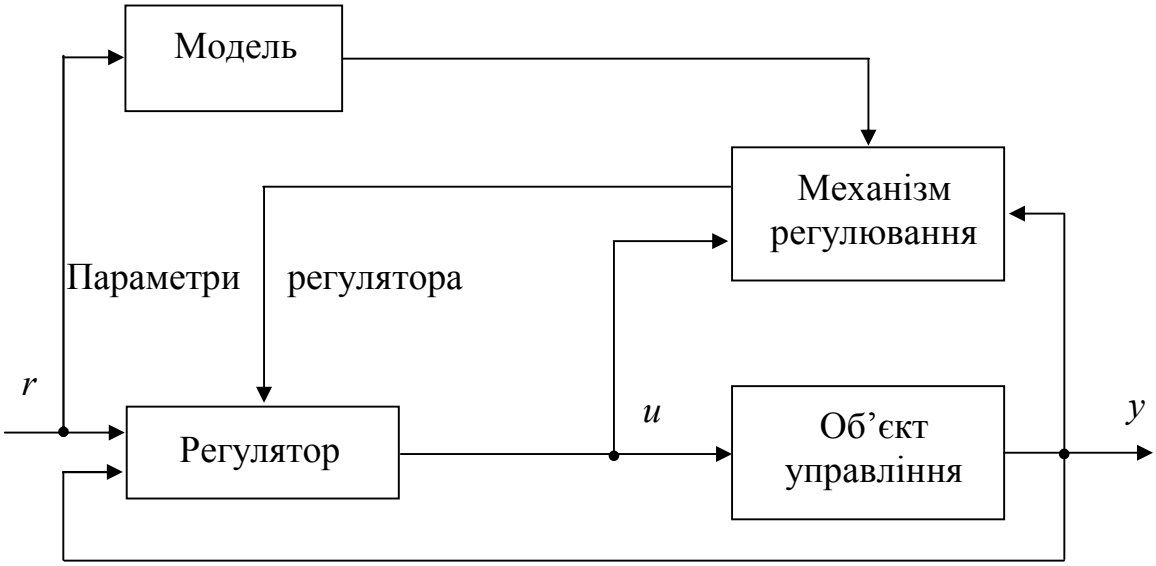


Рисунок 1.6 – Блок-схема адаптивного управління з використанням еталонної моделі

Одним із перших методів побудови нейромережевих СУ був підхід, заснований на «копіюванні» існуючого контролера. У 1964 році Уїдроу запропонував цей метод як спосіб створення експертної системи шляхом отримання знань від вже наявного експерта. Архітектура такої системи показана на рис. 1.7.

Хоча на перший погляд може здатися, що такий підхід надлишковий, він має певні переваги: по-перше, існуючий контролер може бути складним або незручним для використання (наприклад, коли роль контролера виконує людина), а по-друге, ШНМ дозволяє використовувати більш зручні для обчислень та формалізації форми представлення інформації про стан об’єкта.

Сучасні нейромережеві СУ включають широкий спектр архітектур, добре розроблених і застосовуваних у практиці. Усі нейромережеві контролери спрямовані на формування керуючого сигналу, який переводить об’єкт із початкового стану в бажаний, слідуючи оптимальній траєкторії. Спосіб організації контролю залежить від вибору структури та алгоритму навчання. Найпоширені-

шими є пряме та непряме управління, зазвичай із застосуванням алгоритму зворотного розповсюдження помилки.

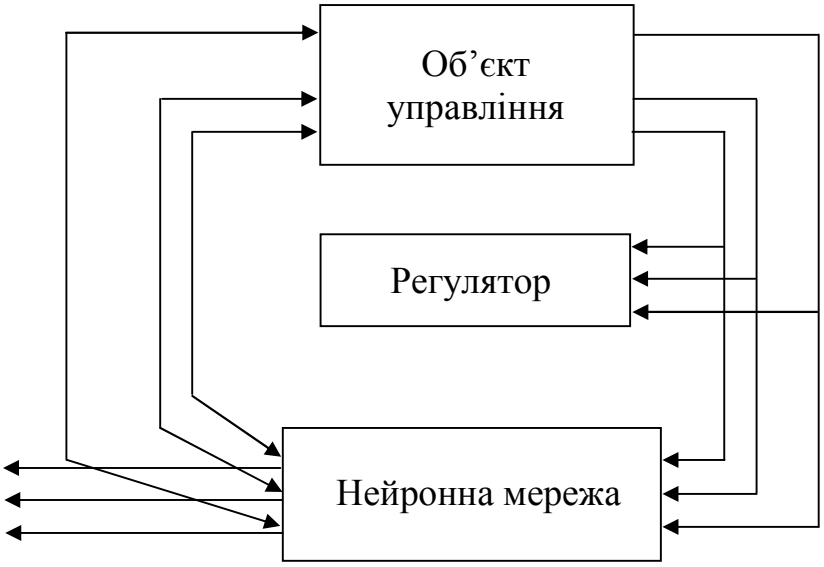


Рисунок 1.7 – Нейромережевий контролер, побудований за принципом «копіювання» існуючого контролера

У схемі непрямого управління параметри об’єкта оцінюються в кожен момент часу та використовуються для регулювання параметрів контролера (рис. 1.8). Основним недоліком такого підходу є те, що ідентифікація та управління базуються виключно на похибці, і гарантії мінімізації вихідної помилки немає. У схемі прямого управління нейромережевий контролер регулює свої параметри для безпосереднього зменшення похибки на виході (рис. 1.9).

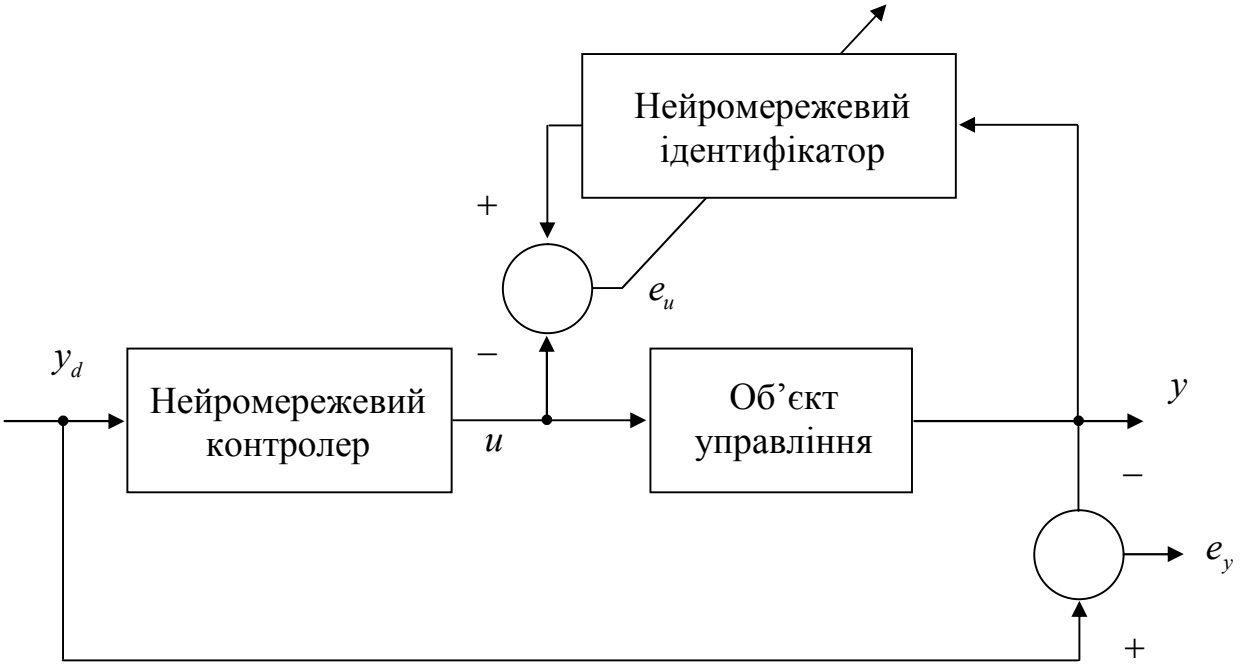


Рисунок 1.8 – Схема непрямого управління

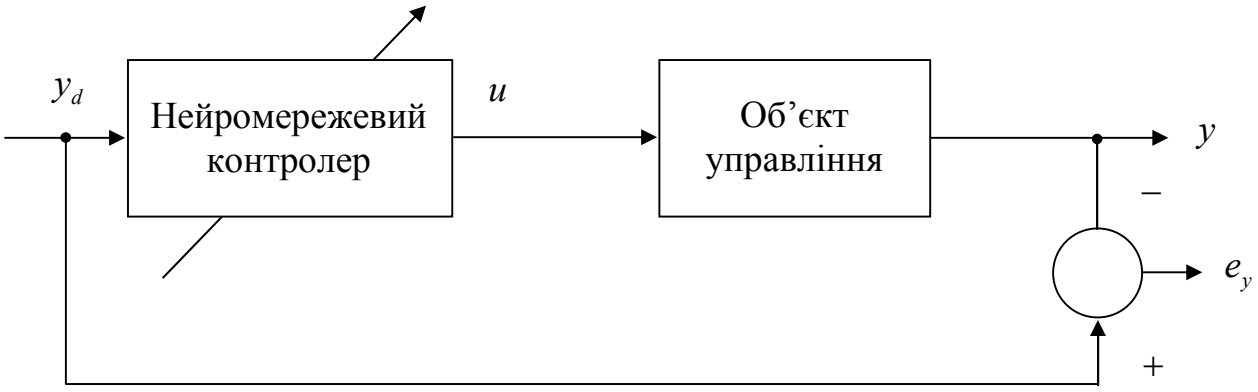


Рисунок 1.9 – Схема прямого управління

Цільовою функцією, яку мінімізує контролер, часто є середньоквадратична помилка на виході об’єкта управління:

$$E_y = \frac{1}{2}(y_d - y)^2. \tag{1.13}$$

Питання стійкості та керованості таких систем детально розглядаються у роботах [53–63].

Принцип предиктивного регулювання на основі моделі (Model Predictive Control, MPC) [64–68] (рис. 1.10) полягає у формуванні послідовності сигналів,

які мінімізують розбіжність між заданим значенням та прогнозованим вихідним сигналом у майбутньому (на горизонті прогнозу).

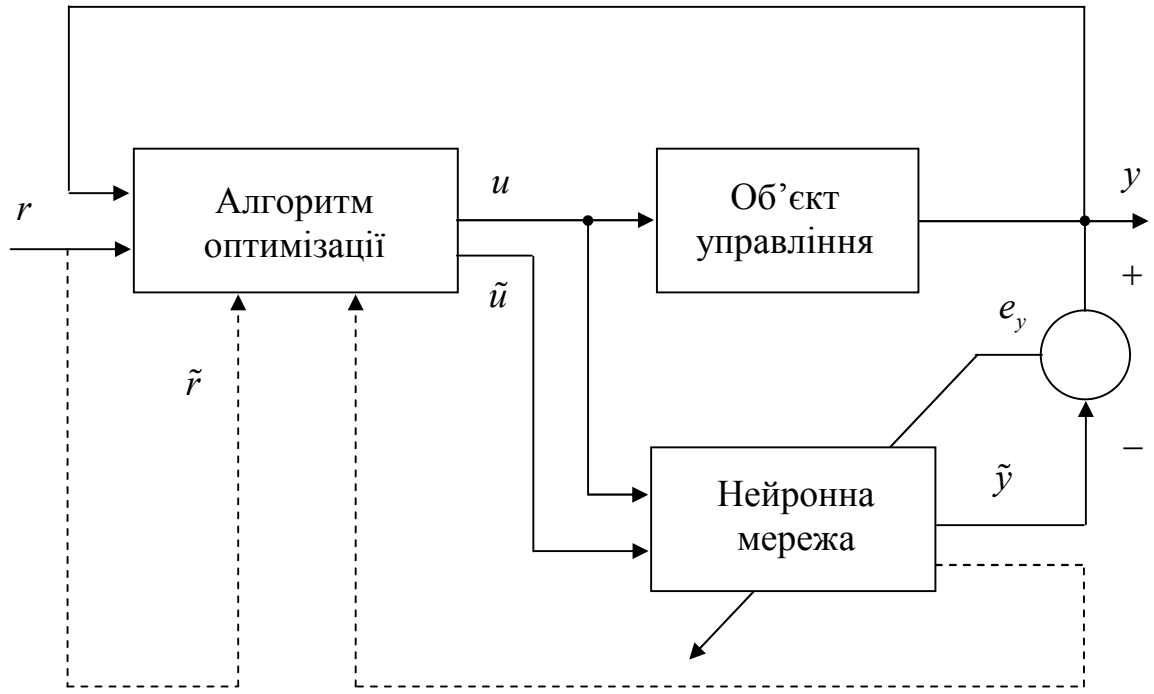


Рисунок 1.10 – Предиктивне регулювання на основі моделі

Регулятор у такій схемі функціонує як ітераційний алгоритм оптимізації:

- оптимізація виконується чисельними методами;
- мінімізується функціонал якості, що описує бажані характеристики системи;
- для впливу на об'єкт використовується лише перший елемент оптимізованої послідовності;
- оптимізація повторюється на кожному кроці дискретизації.

Перевагою MPC є висока якість регулювання навіть для складних нелінійних об'єктів, водночас недоліком залишається велика обчислювальна складність та підвищені вимоги до точності моделі об'єкта. У цій роботі досліджується застосування нелінійного предиктивного регулювання.

Одним із напрямів теоретичних досліджень є порівняння методів ШНМ з класичними СУ, виявлення їхніх специфічних властивостей та проведення аналізу [69]. Порівнювались стійкість, швидкість збіжності, ефективність у шумо-

вих умовах, обсяг необхідної пам’яті тощо. Хоча кожен метод має переваги та недоліки, нейромережеве управління демонструє ряд корисних властивостей, складних для реалізації іншими методами. Основні результати порівняння наведені в табл. 1.1.

Таблиця 1.1 – Порівняння характеристик методів управління

Критерій	Зворотний зв’язок з регульованими коефіцієнтами	Адаптивне управління з еталонною моделлю Ляпунова	Нейромережеве управління
Стійкість зворотного зв’язку	Найгірша	Найкраща	Середня
Швидкість збіжності	Найкраща	Середня	Найгірша
Робота у реальному часі	Середня	Середня	Найкраща
Складність програмного забезпечення	Найгірша	Середня	Середня
Помилка стеження	Середня	Найкраща	Середня
Зменшення перешкод	Найкраща	Найгірша	Середня
Робастність до розузгодження моделі	Найгірша	Середня	Найкраща

1.5 Вимоги до систем управління електроприводами основних механізмів кранів

Мостові крани займають важливе місце на підприємствах машинобудівної галузі, а також у багатьох інших секторах промисловості. Вони призначені для підйому, опускання та переміщення великих заготовок, деталей і вузлів обладнання по цеху як у горизонтальному, так і у вертикальному напрямках. Вибір конкретного типу мостового крана залежить від технологічних особливостей цеху та характеру виробничих процесів, проте більшість конструктивних елементів, таких як механізми підйому та пересування, мають стандартизовану конструкцію та використовуються у різних типах кранів.

В залежності від типу вантажу застосовуються різні пристрої для його захоплення: крюки, магніти, грейфери, кліщі тощо. Найбільш поширеним є використання кранів із підвіскою крюка. Відповідно до чинних стандартів, крани класифікуються за режимами експлуатації механічного та електричного обладнання на чотири категорії: Л – легкий режим, З – середній, Т – важкий, ВТ – дуже важкий. Кожна категорія визначає інтенсивність роботи, навантаження та експлуатаційні умови. Основними показниками режиму роботи є тривалість роботи двигуна механізму (ПВ), кількість включень за годину та коефіцієнти використання механізмів відносно вантажопідйомності.

Механізми мостових кранів функціонують у режимі повторюваних короткочасних циклів з частими включеннями протягом години, часто за складних умов зовнішнього середовища. Тому для них розроблені спеціальні кранівські та металургійські електродвигуни з підвищеною здатністю витримувати перевантаження. Основні механізми для переміщення вантажів у всіх типах кранів включають підйомні лебідки та системи пересування крана й візка. Загальний вигляд мостового крана наведено на рис. 1.13.

Несуча зварна конструкція крана складається з моста, що містить дві головні балки коробчатого перетину або гратчасті ферми, розташовані через проліт цеху, а також кінцеві балки з ходовими колесами. Колеса рухаються по

рейках підкранового шляху, закріплених на опорних балках верхньої частини цеху. Привід коліс забезпечується електродвигунами через редуктори. Уздовж моста прокладено рейки, по яких переміщується візок з підйомною лебідкою, що приводиться в обертання електродвигуном через редуктор. Лебідка оснащена барабаном, на який намотується підйомний трос, що кріпиться до крюка, встановленого на блоках для захоплення вантажу.

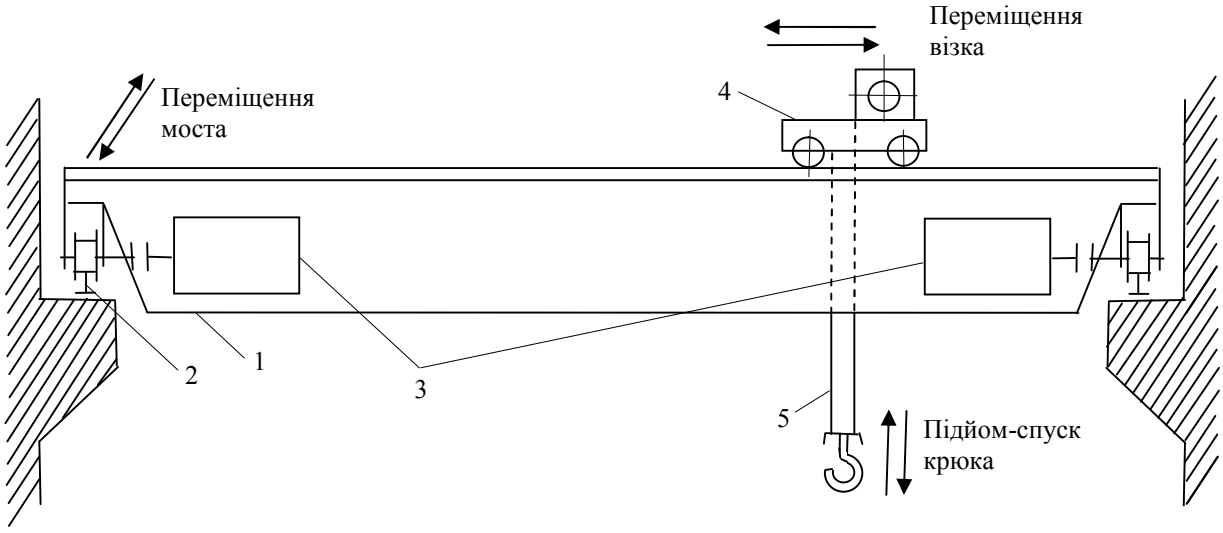


Рисунок 1.13 – Загальний вигляд мостового крана

1 – несуча ферма; 2 – рухомі опори; 3 – механізм пересування моста; 4 – механізм пересування візка; 5 – механізм підйому

Інтенсивне використання кранів і складні умови роботи електроприводів обумовлюють необхідність високого рівня надійності та простоти експлуатації. Тому при проектуванні електроприводів перевага надається простим та зрозумілим схемам управління, навіть якщо це передбачає певне зниження вимог до статичних і динамічних характеристик приводів.

Оскільки керування підйомними кранами здійснюється оператором у режимі реального часу, ключовою технологічною вимогою для систем електроприводу є можливість точного регулювання швидкості. Діапазон регулювання швидкості DDD визначається відношенням робочої швидкості до мінімальної, необхідної для безпечного та плавного виконання операцій. Мінімальна швидкість підйому обмежується умовами м’якого встановлення вантажу без

небажаних поштовхів. При управлінні механізмами пересування регулювання швидкості дозволяє оператору точніше позиціонувати вантаж, зменшуючи кількість повторних включень приводу.

Необхідна точність установки вантажів залежить від призначення крана. Для монтажних кранів, що виконують точні операції, та для кранів, що транспортують крихкі деталі, мінімальна швидкість повинна бути нижчою, ніж у універсальних кранів. Таким чином, діапазон регулювання швидкості у спеціалізованих кранах зазвичай більший і збільшується зі зростанням робочої швидкості механізму.

Регулювання швидкості електроприводу повинно враховувати обмеження по прискоренню, струму та моменту. У більшості кранів робочі швидкості основних механізмів такі, що при допустимих середніх прискореннях перехідні процеси займають невелику частину загального циклу роботи. Тому в цих умовах регулювання моменту і точне дотримання законів руху не впливають суттєво на продуктивність. Водночас при проєктуванні необхідно враховувати вплив коливань вантажу, кінематичних зазорів та інших факторів, і при необхідності застосовувати додаткові заходи для обмеження динамічних навантажень.

1.6 Обґрунтування вибору системи електроприводу механізму підйому мостового крана

Система управління електроприводом є складним комплексом, що включає різноманітні компоненти та вузли. До її складу входять перетворювач електричної енергії, пристрої управління для комутації струму в колі електродвигуна, засоби ручного або автоматичного керування, а також елементи захисту електрообладнання і механізми впливу на пристрої відключення електроприводу.

Електричні ланцюги в таких системах можна класифікувати за функціональним призначенням:

1. Головні ланцюги – забезпечують основний потік енергії до електроприводу та живлення вантажопідйомних магнітів.

2. Ланцюги збудження – через них проходить струм збудження для двигунів постійного струму, синхронних машин змінного струму або електромагнітів гальмівних систем, а також для двигунів електрогідравлічних товкачів.

3. Ланцюги управління – передають команди від органів керування до комутаційних пристроїв головних і збуджувальних ланцюгів, забезпечуючи виконання послідовності операцій і перемикань згідно із заздалегідь визначеною програмою.

4. Ланцюги сигналізації – інформують оператора або контрольний пристрій про стан комутаційних елементів та параметри електроприводу.

У кранових електроприводах застосовуються два основні типи перетворювачів електричної енергії: електромашинні та статичні. Електромашинні перетворювачі включають дві або більше електричних машин, що перетворюють електроенергію джерела живлення в енергію з регульованими параметрами (напругою, струмом, частотою). Статичні перетворювачі здійснюють перетворення електричної енергії за допомогою безконтактної комутації ланцюгів постійного або змінного струму із застосуванням керованих та некерованих напівпровідникових приладів.

Система управління електроприводом включає як контактні, так і безконтактні пристрої комутації, перетворювачі енергії, елементи керування та захисту. Контактні пристрої можна поділити на дві групи:

1. Апаратура, що управляється оператором або виконавчим механізмом (контролери, кінцеві вимикачі).

2. Апаратура, де комутація здійснюється електромагнітними засобами (контактори, реле).

Якщо контактні елементи призначені для перемикання основних електричних ланцюгів і приводяться в дію вручну, вони називаються силовими кулачковими контролерами. Для комутації ланцюгів управління використовують-

ся командоконтролери. При активації контактів через механізм вони називаються кінцевими або путніми вимикачами. Послідовність включення та вимикання контактів за допомогою вала з кулачковими шайбами називається діаграмою включень, яка в табличній формі відома як таблиця включень. Об'єднання кількох контактів, реле та захисних пристроїв в одному комплектному пристрої для керування електроприводом формує магнітний контролер. Якщо перемикання відбувається без розриву електричного кола під напругою, він називається магнітним контролером з бездуговою комутацією.

Оцінка ефективності систем управління повинна враховувати мінімізацію витрат на придбання, обслуговування та ремонт обладнання, а також енергоспоживання під час запуску та зупинки кранів протягом усього терміну експлуатації до капітального ремонту (близько 10 років). Вибір системи здійснюється на основі економічної доцільності.

Якщо економічні показники порівнюваних систем близькі, додатково оцінюються габарити та умови розміщення електрообладнання. У ряді випадків технічні обмеження щодо розташування перетворювальних агрегатів або напівпровідникових перетворювачів можуть вимагати вибору системи з трохи меншими економічними показниками.

Після аналізу технічних характеристик і експлуатаційних можливостей наявних електроприводів для кранівських механізмів, оптимальним вибором для механізму підйому мостового крана є система ТП-Д.

1.7 Постановка задачі дослідження і шляхи її вирішення

Аналіз сучасної наукової літератури показує, що існує велика кількість методів синтезу систем управління, проте універсального регулятора, який задовольняв би всі вимоги до якості управління, на сьогодні не створено. При розробці систем управління механізмом підйому мостового крана слід враховувати пружність підйомного канату, яка, як було показано в главі 3, викликає коливальний характер перехідних процесів і негативно впливає на динаміку робо-

ти механізму. Традиційні регулятори не завжди забезпечують прийнятні показники ефективності системи.

Сучасна теорія адаптивного та оптимального управління дозволяє створювати системи, що функціонують із метою мінімізації або максимізації заздалегідь визначеного критерію якості. Вона включає не лише методи синтезу управлінських пристроїв, а й алгоритми ідентифікації та оцінки станів об'єкта. Завдяки цим алгоритмам можна визначати структуру моделі об’єкта та відновлювати її параметри як під час роботи системи, так і поза нею. Такі підходи застосовуються як у класичних, так і в адаптивних системах управління.

Класична теорія управління базується на розвиненій теорії лінійних систем. Її методи часто застосовуються при дослідженні та синтезі управлінських систем для нелінійних об'єктів, однак оптимальні результати досягаються лише тоді, коли нелінійність об'єкта незначна або об’єкт має великі постійні часи та стабільний розімкнений стан. Управління динамічними об’єктами ускладнюється необхідністю постійного контролю не лише параметрів моделі, а й іноді її структури. Для апроксимації нелінійностей застосовуються ряди Вольтерра, Гаммерштейна, Вінера, поліноми Колмогорова-Габора та інші підходи. Проте використання таких моделей має обмеження, оскільки попередньо необхідно визначити порядок моделі та провести структурну ідентифікацію. Крім того, зовнішні збурення та похибки вимірювань ускладнюють процес управління, а робота з динамічними об'єктами в реальному часі вимагає швидкодіючих алгоритмів.

Незважаючи на численні дослідження, різноманітність типів нелінійностей, динамічні властивості об’єктів, їх нестаціонарність, а також зовнішні збурення й похибки вимірювань не дозволяють розробити універсальний підхід до побудови систем управління. Класичні та сучасні методи часто спираються на лінеаризацію, однак у практичних задачах така апроксимація не завжди забезпечує достатній рівень якості управління, оскільки лінійна модель може не відображати реальні властивості системи.

Штучні нейронні мережі (ШНМ) є перспективною альтернативою класичним методам управління нелінійними об’єктами. Вони належать до галузі штучного інтелекту та використовують принципи, подібні до роботи нейронів живих організмів. Основною перевагою ШНМ є паралельна обробка інформації всіма елементами, що дозволяє виконувати обчислення у реальному часі. Велика кількість міжнейронних зв’язків робить систему стійкою до локальних помилок: пошкоджені зв’язки можуть бути замінені іншими, зберігаючи працездатність мережі.

Ще однією важливою властивістю нейронних мереж є здатність до навчання та узагальнення отриманих знань. Навчившись на обмеженому наборі даних, мережа здатна демонструвати коректні результати на нових вхідних даних, що не були використані під час навчання. Завдяки наявності нейронів з нелінійними функціями активації ШНМ ефективно застосовуються для управління нелінійними динамічними об’єктами, де традиційні методи не забезпечують потрібного рівня точності.

Сучасні системи управління активно використовують такі нейронні архітектури, як багатошаровий персептрон, радіально-базисні та нейро-фаззі мережі. Ці структури здатні апроксимувати будь-які функції, проте для їх навчання потрібна значна кількість даних у форматі «вхід-вихід» і правильний вибір нейромоделі (кількість шарів, нейронів у шарах та тип функцій активації). Для моделювання нелінійних залежностей застосовуються сигмоїдальні та радіально-базисні функції активації.

На відміну від традиційних методів оптимального та адаптивного управління, що потребують великого обсягу апріорної інформації, нейронні мережі здатні до самонавчання. Це дозволяє використовувати нейрорегулятори навіть у умовах значних невизначеностей і обмеженої інформації про об'єкт управління.

Розробка системи управління на основі нейронної мережі передбачає два етапи:

1. Ідентифікація об'єкта управління – створення нейромоделі механізму підйому на основі навчальних пар вхід-вихід, отриманих із вимірюваних значень.

2. Розробка алгоритму управління – синтез регулятора відповідно до заданих цілей та критеріїв якості управління.

Мета роботи: створення нейромережевої системи управління механізмом підйому мостового крана з урахуванням пружних властивостей підйомного канату.

Завдання для досягнення мети:

1. Вибір системи електроприводу механізму підйому, електродвигуна та основного електрообладнання силового кола; синтез системи управління електроприводом та її моделювання на ЕОМ.

2. Розробка математичної моделі системи з урахуванням пружних властивостей канату, моделювання багатомасової системи та аналіз результатів.

3. Створення структурної схеми нейромережевої системи управління для забезпечення високоякісного регулювання із врахуванням пружності механічних зв’язків.

4. Синтез моделі багатомасової системи за допомогою багатошарової прямонаправленої нейронної мережі з високою точністю ідентифікації.

5. Розробка нейромережевого регулятора на основі предикативного методу з використанням нейромоделі об’єкта для забезпечення високої швидкодії та якості управління.

6. Моделювання нейромережевої системи та аналіз отриманих результатів.

ВИСНОВКИ ДО РОЗДІЛУ 1

1. Аналіз наукових джерел підтвердив доцільність використання штучних нейронних мереж у задачах управління. Сучасні системи керування складними динамічними об’єктами повинні мати здатність ефективно адаптуватися до змін умов експлуатації шляхом оперативного коригування параметрів і структури законів управління. Ці вимоги можуть бути реалізовані за допомогою апарату теорії адаптивного управління. Одним із ефективних підходів є застосування методів нейромережевого моделювання та управління, що дозволяють виконувати ідентифікацію та регулювання як на етапі проектування системи, так і під час її функціонування.

2. Дослідження показали, що багатошарові мережі прямого розповсюдження сигналу ефективно застосовуються для побудови нейромережевих систем управління. Двошарова мережа здатна досить точно апроксимувати будь-яку функцію з обмеженою кількістю розривів, за умови достатньої кількості нейронів у прихованому шарі. У роботі обговорено питання навчання нейронних мереж та обрано оптимальний метод навчання для нейромережевого регулятора.

3. Створення адекватних математичних моделей динамічних систем є ключовим етапом у задачах управління, прогнозування та моделювання. Дослідження підтвердили, що ідентифікацію нелінійних динамічних об’єктів ефективно здійснювати за допомогою штучних нейронних мереж. Найчастіше для цього застосовуються багатошаровий персептрон та радіально-базисні мережі.

4. На основі аналізу сучасних структур систем нейрорегулювання обрано принцип нелінійного предиктивного регулювання як базовий підхід для створення нейромережевого регулятора.

5. Проведено аналіз вимог до систем управління електроприводами основних механізмів кранів. Електроприводи кранів працюють у складних умовах і мають широку сферу застосування, тому до них висуваються високі вимоги щодо надійності та простоти експлуатації. Основною технологічною вимогою,

що визначає вибір системи електроприводу, є можливість точного регулювання швидкості, оскільки оператор здійснює управління механізмами на всіх етапах роботи. Регулювання швидкості повинно виконуватися з урахуванням обмежень по прискоренню, струму та моменту. При проектуванні необхідно враховувати вплив розгойдування вантажів, кінематичних зазорів та інших факторів, а за потреби впроваджувати додаткові заходи для зменшення динамічних навантажень.

6. Сформульовано завдання дослідження, яке полягає у розробці нейромережевої системи управління багатомасовою електромеханічною системою механізму підйому мостового крана, що забезпечує високу якість регулювання.

РОЗДІЛ 2
РОЗРАХУНОК СИСТЕМИ АВТОМАТИЧНОГО УПРАВЛІННЯ МЕХАНІЗМОМ ПІДЙОМУ МОСТОВОГО КРАНА

2.1 Технічні характеристики мостового крана та вимоги до системи управління

Для розрахунку електроприводу механізму підйому мостового крана та синтезу системи автоматичного управління використані вихідні дані, наведені у таблиці 2.1.

Таблиця 2.1 – Основні технічні характеристики 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 роботи системи управління.

Система управління повинна бути астатичною. Перерегулювання швидкості при ступінчастій вхідній дії не повинно перевищувати 10 %. Час регулювання системи не має перевищувати 5 секунд.

2.2 Розрахунок потужності електроприводу механізму підйому та вибір приводного двигуна

Статична потужність, що необхідна на валу двигуна, визначається за формулою:

$$P_{\text{ст.п}} = \frac{q(G_{\text{вп}} + m_0)V_{\text{п}}10^{-3}}{\eta_{\text{н.п}}},$$

де q – прискорення вільного падіння $q = 9{,}81\ м/с^2$;

$G_{\text{вп}}$ – маса вантажу (вантажопідйомність крана);

m_0 – маса вантажозахватного пристрою;

$V_{\text{п}}$ – швидкість підйому вантажу;

$\eta_{\text{н.п}}$ – сумарний коефіцієнт корисної дії канатної системи та механізму підйому:

$$\eta_{\text{н.п}} = \eta_{\text{к.п}} \cdot \eta_{\text{к}} \cdot \mu_{\text{р}},$$

де $\eta_{\text{к.п}}$ – коефіцієнт корисної дії канатної передач $\eta_{\text{к.п}} = 0{,}98$;

$\eta_{\text{к}}$ – ККД канатоукладчика $\eta_{\text{к}} = 0{,}98$;

$\eta_{\text{р}}$ – ККД редуктора $\eta_{\text{р}} = 0{,}84$;

Ця формула враховує сумарні втрати енергії на всіх елементах механізму підйому, що дозволяє правильно розрахувати необхідну потужність двигуна.

$$\eta_{\text{н.п}} = 0{,}98 \cdot 0{,}98 \cdot 0{,}84 = 0{,}8,$$

$$P_{\text{ст.п}} = \frac{9{,}81 \cdot (37000 + 1000) \cdot 0{,}1 \cdot 10^{-3}}{0{,}8} = 47\ кВт.$$

Число циклів підйому визначається як:

$$Z = \frac{Q}{G_{\text{гр}}},$$

де Q – продуктивність крана:

$$Z = \frac{55}{37000} = 1,486 \cdot 10^{-3}.$$

Час циклу

$$T = \frac{1}{Z},$$

$$T = \frac{1}{1,486 \cdot 10^{-3}} = 670\ c.$$

Орієнтовний час роботи механізму підйому

$$t_{\text{р.п}} = 4 \cdot \frac{L_{\text{п}}}{V_{\text{п}}},$$

де $L_{\text{п}}$ – середня висота підйому та спуску вантажу;

$$t_{\text{р.п}} = 4 \cdot \frac{6}{0,1} = 240\ c.$$

Відносна тривалість включення двигуна:

$$ПВ_{\text{п}} = \frac{t_{\text{р.п}}}{T} \cdot 100\%,$$

$$ПВ_{\text{п}} = \frac{240}{670} = 0,36.$$

Розрахункова потужність двигуна, приведена до стандартної тривалості включення:

$$P_{\text{п}} = P_{\text{ст.п}} \cdot \sqrt{\frac{ПВ_{\text{п}}}{ПВ_{\text{ст}}}}.$$

де $ПВ_{\text{ст}}$ – стандартна тривалість включення; для мостових кранів приймається $ПВ_{\text{ст}} = 40\%$.

$$P_{\text{п}} = 47 \cdot \sqrt{\frac{0,36}{0,4}} = 44,6 \ кВт .$$

Відповідно до розрахунків, для механізму підйому обрано електродвигун типу Д812 з наступними параметрами:

$P_{\text{н}} = 45 \ кВт$; $n_{\text{н}} = 570 \ об/хв$; $I_{\text{н}} = 92 \ А$; $U_{\text{н}} = 440 \ В$; $R_{\text{я}} + R_{\text{д.п}} = 0,1 \ Ом$; $N = 418$; $a = 2$; $p = 2$; $\Phi_{\text{н}} = 50,1 \cdot 10^{-3} \ Вб$; ПВ=40%; $J_{\text{дв}} = 7 \ кг \cdot м^2$

Номінальна швидкість обертання обраного двигуна:

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

$$\omega_{\text{н}} = \frac{3,14 \cdot 570}{30} = 59,7 \ об/хв .$$

2.3 Перевірка двигуна по нагріву і перевантажувальній спроможності

Передавальне число механізму підйому визначається як:

$$i_{\text{п}} = \frac{\omega_{\text{н}} D}{2V_{\text{п}}},$$

де D – діаметр барабана механізму підйому,

$\omega_{\text{н}}$ – кутова швидкість валу двигуна:

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

$$\omega_{\text{н}} = \frac{3,14 \cdot 570}{30} = 59,7 \ об/хв ,$$

$$i_{\text{п}} = \frac{59,7 \cdot 0,65}{2 \cdot 0,1} = 194 .$$

Щоб підтвердити правильність вибраного двигуна, виконаємо побудову швидкісної діаграми та діаграми навантаження.

У статичному режимі під час підйому вантажу крутний момент на валу підйомного двигуна визначається як:

$$M_{\text{ст.п}} = \frac{(G_{\text{вп}} + m_0)qD}{2i_{\text{п}}\eta_{\text{н.п}}}.$$

$$M_{\text{ст.п}} = \frac{(37000 + 1000) \cdot 9{,}81 \cdot 0{,}65}{2 \cdot 194 \cdot 0{,}8} = 781 \text{ Нм}.$$

Момент підйому порожнього вантажозахватного пристрою обчислюється з урахуванням коефіцієнта ККД $\eta_{0.\text{п}}$ передачі для порожнього вантажу:

$$M_{\text{ст.п.0}} = \frac{m_0 qD}{2i_{\text{п}}\eta_{0.\text{п}}}.$$

Для отримання необхідного значення $\eta_{0.\text{п}}$ визначаємо коефіцієнт навантаження під час підйому порожнього вантажозахоплювального пристрою:

$$\gamma_{\text{п}} = \frac{m_0}{G_{\text{вп}} + m_0},$$

$$\gamma_{\text{п}} = \frac{1000}{37000 + 1000} = 0{,}03.$$

Обчислюємо коефіцієнт корисної дії приводу під час підйому порожнього вантажозахоплювача: $\eta_{0.\text{п}} = 0{,}28$.

$$M_{\text{ст.п.0}} = \frac{1000 \cdot 9{,}81 \cdot 0{,}65}{2 \cdot 194 \cdot 0{,}28} = 59 \text{ Нм}.$$

Крутний момент підйомного двигуна в статичному режимі під час опускання максимального вантажу обчислюється як:

$$M_{\text{ст.с}} = \frac{(G_{\text{вп}} + m_0)q}{2i_{\text{п}}} D\left(2 - \frac{1}{\eta_{\text{н.п}}}\right),$$

$$M_{\text{ст.с}} = \frac{(37000 + 1000) \cdot 9{,}81}{2 \cdot 194} \cdot 0{,}65 \cdot \left(2 - \frac{1}{0{,}8}\right) = 468\ Н \cdot м.$$

Крутний момент двигуна в статичному режимі при опусканні порожнього вантажозахоплювача:

$$M_{\text{ст.с.0}} = M_{\text{ст.п.0}} - \frac{M_{\text{ст.п.0}} - M_{\text{ст.с}}}{2 \cdot m_0 + G_{гр}} \cdot 2m_0.$$

$$M_{\text{ст.с.0}} = 59 - \frac{59 - 468}{2 \cdot 1000 + 37000} \cdot 2 \cdot 1000 = 80\ Нм.$$

Приведені моменти інерції на валу двигуна під час підйому та спуску вантажу:

$$J_{\text{пр.п(с)}} = \delta J_{\text{дв}} + \frac{(G_{\text{вп}} + m_0)V_{\text{п}}^2}{\omega_{\text{н}}^2},$$

де δ – коефіцієнт, що враховує додаткову інерцію передач, δ=1,2.

$$J_{\text{пр.п(с)}} = 1{,}2 \cdot 7 + \frac{(37000 + 1000) \cdot 0{,}1^2}{59{,}7^2} = 8{,}4 + 0{,}11 = 8{,}51\ кг \cdot м^2.$$

Для порожнього вантажозахватного пристрою використовуються аналогічні формули:

$$J_{\text{пр.0}} = \delta J_{\text{дв}} + \frac{m_0 + V_{\text{п}}^2}{\omega_{\text{н}}^2},$$

$$J_{\text{пр.0}} = 1{,}2 \cdot 7 + \frac{1000 \cdot 0{,}1^2}{59{,}7^2} = 8{,}4 + 0{,}005 = 8{,}405\ кг \cdot м^2.$$

Розрахунок тривалості розгону та зупинки електропривода підйомного механізму мостового крана.

Час розгону й гальмування визначаємо з урахуванням допустимих значень прискорень $a_{\text{п}} = 0{,}1\ м/с^2$. Зважаючи на те, що зміна швидкості на природній характеристиці є незначною, для всіх робочих режимів механізму підйому приймаємо:

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

$$t_{\text{п}} = t_{\text{т}} = \frac{0{,}1}{0{,}1} = 1\ c.$$

Відповідний шлях вантажу під час прискорення або гальмування:

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

$$l_{\text{т}} = l_{\text{п}} = \frac{0{,}1}{2} \cdot 1 = 0{,}05\ м.$$

Час руху із сталою швидкістю обчислюється як:

$$t_{\text{у}1} = t_{\text{у}2} = t_{\text{у}3} = t_{\text{у}4} = \frac{L_{\text{п}} - (l_{\text{т}} + l_{\text{п}})}{V_{\text{п}}},$$

$$t_{\text{у}1} = t_{\text{у}2} = t_{\text{у}3} = t_{\text{у}4} = \frac{6 - (0{,}05 + 0{,}05)}{0{,}1} = 59\ c.$$

Уточнюється тривалість включення механізму:

$$ПВ_{\text{п}} = \frac{\Sigma t_{\text{п}} + \Sigma t_{\text{т}} + \Sigma t_{\text{у}}}{T},$$

$$ПВ_{\text{п}} = \frac{4 \cdot 1 + 4 \cdot 1 + 2 \cdot 59 + 2 \cdot 59}{670} = 0{,}364.$$

Для перевірки вибору двигуна розрахунок діаграми навантаження електроприводу підйому мостового крана виконується по ділянках циклу роботи.

1. Підйом вантажу номінальної маси.

1.1 Розгін під час підйому номінального вантажу

$$M_{\text{дв п р}} = M_{\text{ст.п}} + M_{\text{дин.п}} \ ,$$

де $M_{\text{дин.п}}$ – динамічний крутний момент, що виникає під час розгону при підйомі вантажу

$$M_{\text{дин.п}} = J_{\text{пр.п(с)}} \cdot \frac{\omega_{\text{н}}}{t_{\text{п}}},$$

$$M_{\text{дин.п}} = 8,51 \cdot \frac{59,7}{1} = 502 \ Нм \ ;$$

$$M_{\text{дв п р}} = 781 + 502 = 1283 \ Нм \ .$$

1.2 Рух із постійною швидкістю під час підйому вантажу номінальної маси.

$$M_{\text{дв п у}} = M_{\text{ст.п}} = 781 \ Нм \ .$$

1.3 Гальмування при підйомі вантажу номінальної маси:

$$M_{\text{дв п т}} = M_{\text{ст.п}} - M_{\text{дин.п}} \ ,$$

$$M_{\text{дв п т}} = 781 - 502 = 279 \ Нм \ .$$

2. Спуск вантажу номінальної маси.

2.1 Розгін під час спускання номінального вантажу

$$M_{\text{дв с р}} = M_{\text{ст.с}} - M_{\text{дин.с}} \ ,$$

де $M_{\text{дин.с}}$ – динамічний крутний момент, що виникає під час розгону при спуску вантажу

$$M_{\text{дин.с}} = M_{\text{дин.п}} = 502 \ Нм \ ,$$

$$M_{\text{дв с р}} = 468 - 502 = -34 \ Нм \ .$$

1.2 Рух із постійною швидкістю під час спуску вантажу номінальної маси.

$$M_{\text{дв с}} = M_{\text{ст.с}} = 468 \ Нм.$$

2.3 Гальмування при спуску вантажу номінальної маси:

$$M_{\text{дв с т}} = M_{\text{ст.с}} + M_{\text{дин.с}}.$$

$$M_{\text{дв с т}} = 468 + 502 = 970 \ Нм.$$

3. Підйом порожнього вантажозахватного пристрою.

3.1 Розгін при підйомі порожнього вантажозахватного пристрою.

$$M_{\text{дв п р 0}} = M_{\text{ст п 0}} + M_{\text{дин п 0}} \ .$$

де $M_{\text{дин п 0}}$ – динамічний крутний момент, що виникає під час розгону при підйомі вантажозахоплювального обладнання:

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

$$M_{\text{дин.п 0}} = 8,405 \cdot \frac{59,7}{1} = 498 \ Нм,$$

$$M_{\text{дв п р 0}} = 59 + 498 = 557 \ Нм.$$

3.2 Рух із постійною швидкістю під час підйому порожнього вантажозахватного пристрою

$$M_{\text{дв п 0}} = M_{\text{ст.п 0}} = 59 \ Нм.$$

3.3 Гальмування при підйомі порожнього вантажозахватного пристрою:

$$M_{\text{дв п т 0}} = M_{\text{ст п 0}} - M_{\text{дин п 0}} \ ,$$

$$M_{\text{дв п т 0}} = 59 - 498 = -439 \ Нм.$$

4. Спуск порожнього вантажозахватного присттрою.

4.1 Розгін при спуску порожнього вантажозахватного пристрою:

$$M_{\text{дв с р 0}} = M_{\text{ст.с 0}} - M_{\text{дин.с 0}},$$

де $M_{\text{дин с 0}}$ – динамічний крутний момент, що виникає що виникає під час розгону при спуску вантажозахватного пристрою:

$$M_{\text{дин.с 0}} = M_{\text{дин.п 0}} = 498 \ Нм,$$

$$M_{\text{дв с р 0}} = 80 - 498 = -418 \ Нм.$$

4.2 Рух із постійною швидкістю під час спуску вантажозахватного пристрою:

$$M_{\text{дв с 0}} = M_{\text{ст.с 0}} = 80 \ Нм$$

4.3 Гальмування при спуску порожнього вантажозахватного пристрою:

$$M_{\text{дв с т 0}} = M_{\text{ст.с 0}} + M_{\text{дин.с 0}},$$

$$M_{\text{дв с р 0}} = 60 + 498 = 558 \ Нм.$$

Перевірка за еквівалентним моментом. Еквівалентний момент розраховується за формулою:

$$M_{\text{екв}} = \sqrt{\frac{\Sigma M^2{}_{\text{п}_i} t_{\text{п}_i} + \Sigma M^2{}_{\text{т}_i} t_{\text{т}} + \Sigma M^2{}_{\text{у}_i} t_{\text{у}_i}}{\beta(\Sigma t_{\text{п}_i} + \Sigma t_{\text{т}_i}) + \Sigma t_{\text{у}_i}}}, \qquad (2.1)$$

де $t_{\text{п}_i}$, $t_{\text{т}_i}$, $t_{\text{у}_i}$ – тривалість роботи механізму на ділянках, що відповідають моментам $M_{\text{п}_i}$, $M_{\text{т}i}$, $M_{\text{у}_i}$;

$M_{\text{п}_i}$, $M_{\text{т}i}$, $M_{\text{у}_i}$ – середнє значення моментів під час розгону, гальмування та руху з усталеною швидкістю на *i*-тій фазі робочого циклу;

$\beta = \frac{1+\beta_0}{2} = \frac{1+0,35}{2} = 0,675$ – коефіцієнт, що враховує погіршення умов вентиляції під час пуску та гальмування;

$\beta_0 = 0,35$ – коефіцієнт, який відображає зниження ефективності охолодження в період паузи.

Електродвигун вважається придатним, якщо виконується умова: $M_{н} \geq M_{екв\ пр}$, де

$$M_{екв\ пр} = M_{екв} \cdot \sqrt{\frac{ПВ_{п}}{ПВ_{ст}}},$$

В результаті підстановки всіх значень моментів і тривалостей циклу в (2.1) маємо:

$$M_{екв} = 489\ Нм;$$

$$M_{экв\ пр} = 489 \cdot \sqrt{\frac{0,364}{0,4}} = 466\ Нм.$$

Номінальний крутний момент обраного двигуна:

$$M_{н} = \frac{P_{н}}{\omega_{н}},$$

$$M_{н} = \frac{45000}{59,7} = 754\ Нм.$$

Нерівність $M_{н} \geq M_{екв\ пр}$ виконується, отже двигун задовольняє вимогам по нагріву.

Максимальний крутний момент, який розвиває двигун під час розгону привода з номінальним вантажем, становить $M_{max} = M_{дв\ п\ р} = 1283\ Нм$, що менше подвійного перевантаження двигуна. Отже, за показником перевантажувальної здатності двигун також відповідає встановленим вимогам

2.4 Вибір основного електроустаткування електроприводу

Головним параметром при виборі тиристорного перетворювача є його потужність. Вона повинна перевищувати потужність обраного двигуна постійного струму на 5–10 %, щоб забезпечити надійну роботу системи. Крім того, випрямлений струм перетворювача має бути рівним або трохи більшим за номінальний струм двигуна. Напруга перетворювача визначається з урахуванням динамічного запасу, який необхідний для форсування процесу зміни струму, що важливо для реалізації оптимальної настройки контура струму.

Електрорушійна сила (ЕРС) перетворювача визначається за формулою:

$$E_{\text{пр}} = E_{\text{дв н}} + \Delta U ,$$

де $E_{\text{дв н}}$ – номінальна ЕРС двигуна;

ΔU – коефіцієнт запасу по напрузі тиристорного перетворювача.

Номінальна ЕРС двигуна обчислюється як:

$$E_{\text{дв н}} = U_{\text{н}} - I_{\text{н}} R_{\text{я к}} ,$$

де $U_{\text{н}}$, $I_{\text{н}}$ – номінальна напруга та номінальний струм двигуна;

$R_{\text{я к}}$ – опір якірного кола, визначений для нагрітого стану машини за формулою:

$$R_{\text{я к}} = \alpha_t \cdot (R_{\text{я}} + R_{\text{д.п}}) ,$$

де $R_{\text{я}}$, $R_{\text{д.п}}$ – активні опори обмотки якоря та додаткових полюсів;

α_t – коефіцієнт, що враховує збільшення опору при нагріванні. $\alpha_t = 1{,}36$:

$$R_{\text{я к}} = 1{,}36 \cdot 0{,}1 = 0{,}136 \text{ Ом} .$$

$$E_{\text{дв н}} = 440 - 92 \cdot 0{,}136 = 427{,}5 \text{ В} .$$

Запас напруги тиристорного перетворювача визначається як:

$$\Delta U = I_{\text{дв}} R_{\Sigma} + L_{\Sigma} \frac{dI_{\text{дв}}}{dt} \cong k_{\text{ф}} I_{\text{н}} R_{\Sigma},$$

$I_{\text{дв}}$ – струм двигуна;

L_{Σ}, R_{Σ} – сумарний активний опір та індуктивність силового кола електроприводу;

$k_{\text{ф}}$ – коефіцієнт форсування.

Приймаємо наступні значеннгя величин k_{Φ} та R_{Σ}:

$$k_{\text{ф}} = 2, \quad R_{\Sigma} \approx 2R_{\text{я к}} = 2 \cdot 0{,}136 = 0{,}272 \ Ом,$$

і обчислюємо значення ΔU та $E_{\text{дв н}}$:

$$\Delta U = 2 \cdot 92 \cdot 0{,}272 = 50 \ B,$$

$$E_{\text{пр}} = 427{,}5 \ + 50 = 477{,}5 \ B.$$

За каталогом обрано тиристорний перетворювач ТПЕ 100/100-460, з номінальною випрямленою напругою та номінальним випрямленим струмом $I_{\text{ном}} = 100 A$.

Тиристорний перетворювач живиться від мережі 380B, 50 $Гц$ через токообмежувальний реактор. Випрямна схема виконана за трифазною мостовою конфігурацією з двома зустрічно-паралельними групами. До складу перетворювача входить також регульований випрямляч для ланцюгів збудження

Для обмеження пускового струму використовується реактор РТСТ-165-0,084УЗ з такими характеристиками: номінальний струм: $I_{\text{р н}} = 265 A$, індуктивність фази: $L_{\text{р}} = 0{,}156 \cdot 10^{-3} Гн$.

2.5 Розрахунок параметрів силового кола електроприводу

Коефіцієнт посилення тиристорного перетворювача визначається як:

$$k_{\text{тп}} = \frac{1{,}36 \cdot E_{\text{пр max}}}{U_y}$$

де $E_{\text{пр max}}$ – максимальна випрямлена ЕРС тиристорного перетворювача;

$U_{\text{у}}$ – максимальне значення напруги керування, приймаємо $U_{\text{у}} = 15\ B$

Максимальна випрямлена ЕРС перетворювача обчислюється за формулою:

$$E_{\text{пр max}} = U_2\sqrt{2} \cdot \frac{m}{\pi} \cdot \sin\frac{\pi}{m},$$

де U_2 – напруга мережі;

m – кількість тактів випрямлення; для трифазної мостової схеми $m = 6$

$$E_{\text{д max}} = 380 \cdot \sqrt{2} \cdot \frac{6}{3{,}14} \cdot \sin\frac{180^\circ}{6} = 569B.$$

Коефіцієнт підсилення тиристорного перетворювача

$$k_{\text{тп}} = \frac{1{,}3 \cdot 569}{15} = 49{,}3$$

Еквівалентний опір тиристорного перетворювача

$$R_{\text{е пр}} = mfL_{\text{р}},$$

m – кількість пульсацій випрямленої напруги перетворювача; для трифазної мостової схеми $m = 6$;

f – частота мережі;

$$R_{\text{е пр}} = 6 \cdot 50 \cdot 0{,}156 \cdot 10^{-3} = 0{,}0468\ Ом.$$

У тиристорних перетворювачах серії ТПЕ з номінальним струмом менше 500 *А* згладжуючі реактори не застосовуються, тому сумарний активний опір силового кола електроприводу визначається як:

$$R_{\Sigma} = R_{\text{я к}} + R_{\text{е пр}},$$

$$R_{\Sigma} = 0,136 + 0,0468 = 0,1828 \ Ом.$$

Індуктивність якорного ланцюга двигуна розраховується за формулою:

$$L_{\text{я к}} = k\frac{U_{\text{н}}}{I_{\text{н}}\omega_{\text{н}}p},$$

k – коефіцієнт, що приймає значення $k = (1 \div 0,25)$ для двигунів з компенсаційною обмоткою і $k = 0,6$ для двигунів бех компенсаційної обмотки;

p – кількість пар полюсі двигуна,

$$L_{\text{я к}} = 0,6 \cdot \frac{440}{92 \cdot 59,7 \cdot 2} = 2,4 \cdot 10^{-3} \ Гн,$$

$$L_{\Sigma} = L_{\text{я к}} + 2 \cdot L_{\text{р}},$$

$$L_{\Sigma} = 2,4 \cdot 10^{-3} + 2 \cdot 0,156 \cdot 10^{-3} = 2,712 \ \cdot 10^{-3} \ Гн.$$

Електромагнітна стала часу силового кола електроприводу:

$$T_{\text{е}} = \frac{L_{\Sigma}}{R_{\Sigma}},$$

$$T_{\text{е}} = \frac{0,002712}{0,1828} = 0,014 \ с.$$

Електромагнітна стала часу якірного кола двигуна:

$$T_a = \frac{L_{\text{я к}}}{R_{\text{я к}}},$$

$$T_a = \frac{2,4 \cdot 10^{-3}}{0,136} = 0,017\ c.$$

Електромеханічна стала часу електроприводу:

$$T_{\text{м}} = J_{\Sigma} \cdot \frac{R_{\Sigma}}{c^2},$$

де J_{Σ} – сумарний момент інерції, $J_{\Sigma} = J_{\text{пр.п}} = 8,51 кг \cdot м^2$;

c – електромагнітна стала двигуна

$$c = \frac{pN}{2\pi a} \Phi_a,$$

де N – кількість активних провідників обмотки якоря, $N = 418$;

a – кількість пар паралельних гілок обмотки якоря:

$$c = \frac{2 \cdot 418}{2 \cdot 3,14 \cdot 1} \cdot 50,1 \cdot 10^{-3} = 7,16,$$

$$T_{\text{м}} = 8,51 \cdot \frac{0,1828}{7,16^2} = 0,022c.$$

Коефіцієнт підсилення двигуна

$$k_{\text{д}} = \frac{1}{c},$$

$$k_{\text{д}} = \frac{1}{7,16} = 0,14.$$

2.6 Вибір структури системи автоматичного управління

Система автоматичного регулювання (САР) механізму підйому мостового крана повинна забезпечувати:

- необхідні механічні характеристики електроприводу в статичному та динамічному режимах;
- високу якість перехідних процесів;
- оптимізацію роботи електроприводу за різними режимами роботи механізму.

Досягнення високих показників регулювання можливе лише при використанні замкнутих систем автоматичного регулювання. Формування якісних перехідних процесів зводиться до мінімізації часу переходу $t_{\text{і}}$ вихідного параметра y у заданих межах перерегулювання $\Delta y_{\max}$. Зокрема, час $t_{\text{і}}$ визначається входженням перерегулювання у 5%-ву зону:

Характер перехідного процесу визначається співвідношенням постійних часу елементів системи. Для забезпечення високої якості регулювання вводять регулятори, що коректують, які компенсують основні інерційності об'єкта.

Основний підхід до побудови САР: складання структурної схеми з урахуванням характеристик окремих ланок; введення регуляторів для корекції динаміки системи.

На практиці для електроприводів мостових кранів застосовуються системи підлеглого регулювання параметрів. У таких системах використовується принцип компенсації основних інерційностей об’єкта за допомогою послідовно включеного регулятора. Число послідовно включених регуляторів відповідає числу регульованих параметрів, а кожний контур складається з об’єкта регулювання та регулятора.

Для електроприводу механізму підйому обрана двоконтурна дворазово інтегруюча система управління, яка має внутрішній контур – регулювання струму та зовнішній контур – регулювання швидкості по ЕРС двигуна.

Системи регулювання швидкості зі зворотним зв'язком по ЕРС застосовуються, коли точність регулювання швидкості не є критичною. Такі електроприводи є типовими для промислових механізмів.

Схема сигналів:

- Сигнал, пропорційний напрузі, надходить від датчика напруги;
- Сигнал, пропорційний струму, надходить від датчика струму, що враховує падіння напруги в якорному ланцюзі;
- Сумування сигналів відбувається на вході ЕРС.

У якості ЕРС використовується ПІ-регулятор, що забезпечує оптимізацію контура за симетричним критерієм. Така система має:

- астатизм другого порядку по задаючій дії;
- астатизм по збурюючому моменту статичного навантаження.

Ця структура забезпечує:

- стабільну роботу електроприводу в умовах змінного навантаження;
- високоякісне регулювання швидкості підйому і спуску вантажу;
- оптимальне співвідношення динамічних характеристик та стабільності системи.

2.7 Синтез послідовного коректуючого пристрою контура струму

Алгоритмічна схема контуру струму системи підлеглого регулювання наведена на рис. 2.1.

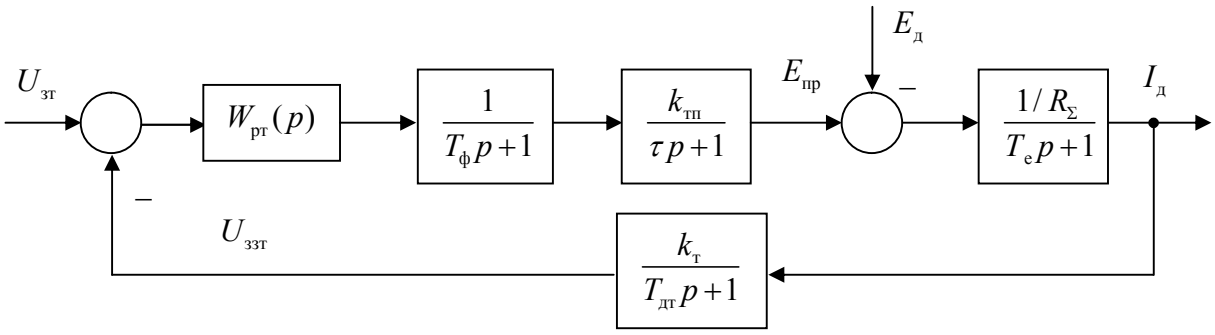


Рисунок. 2.1 – Алгоритмічна схема контура струму системи підлеглого регулювання

Щоб застосувати загальноприйняту методику синтезу послідовного коректуючого пристрою, схему перетворюють до варіанту з одиничним зворотним зв'язком, при цьому зворотний зв'язок по ЕРС двигуна не враховується (рис. 2.2).

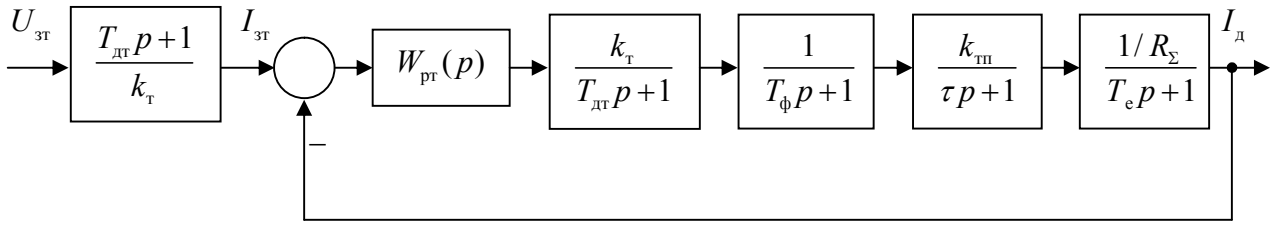


Рисунок. 2.2 – Контур струму з одиничним зворотним зв'язком

Мала некомпенсована постійна часу контуру струму $T_{\mu т}$ визначається як сумарна малих постійних часу його елементів:

$$T_{\mu т} = \tau + T_{дт} + T_ф \ , \qquad (2.2)$$

де τ – середнє затримання тиристорного перетворювача;

$T_{дт}$ – постійна часу датчика струму;

$T_ф$ – постійна часу фільтра вході СІФУ .

Схема з одиничним зворотним зв'язком згортається до розрахункової схеми контуру струму, представленої на рис. 2.3, де форсуюча ланка $(1+T_{дт}p)$ на вході системи не враховується.

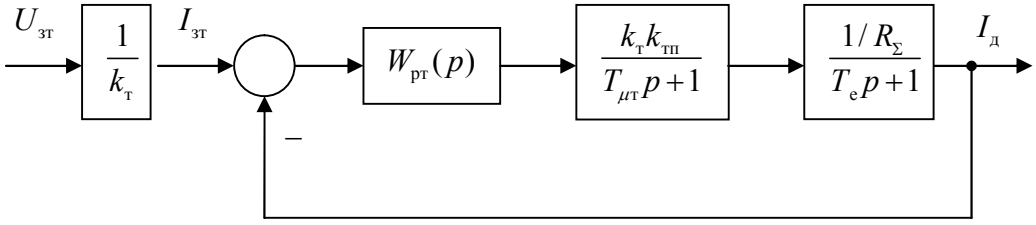


Рисунок 2.3 – – Розрахункова схема контуру струму
для синтезу коректуючого пристрою

Вибираємо $T_{\mu т} = 2,6 \cdot 10^{-3} c$, що знаходиться у практично прийнятих межах 4 ÷ 10 *мс*.

Оптимізацію контура струму здійснюватимемо, застосовуючи модульний критерій, відповідно до якого передавальна функція розімкненого, оптимізованого контуру повинна мати наступний вид:

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

Передавальна функція об'єкта управління контуру струму має вигляд:

$$W_{об\,т}(p) = \frac{k_т \cdot k_{тп}}{(T_{цт} p + 1)} \cdot \frac{1/R_\Sigma}{(T_е p + 1)} \ .$$

Бажана передавальна функція контура струму:

$$W_{бт}(p) = W_{рт}(p) \cdot W_{об\,т}(p),$$

Відповідно, передавальна функція регулятора струму визначається як:

$$W_{рт}(p) = \frac{W_{бт}(p)}{W_{об\,т}(p)} = \frac{\dfrac{1}{2T_{\mu т} p(1 + T_{\mu т} p)}}{\dfrac{k_т \cdot k_{тп}}{(T_{цт} p + 1)} \cdot \dfrac{1/R_\Sigma}{(T_е p + 1)}} = k_{рт} \cdot \frac{T_{рт} p + 1}{T_{рт} p} \ , \qquad (2.3)$$

де $k_{рт}$ - коефіцієнт посилення регулятора;

$T_{рт}$ – постійна часу регулятора:

$$k_{рт} = \frac{T_a}{2T_{\mu т}(1/R_\Sigma) k_т k_{тп}} \ ,$$

$$T_{рт} = T_е \ ;$$

$k_т$ – коефіцієнт підсилення ланцюга зворотного зв'язку по струму:

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

Величину $k_{\text{т}}$ визначаємо з

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

$I_{\text{д max}}$ – максимальний струм якірного ланцюга приймаємо як**:**

$$I_{\text{ш}} = I_{\text{д max}} = 2,5 I_{\text{н}}$$

$$I_{\text{д max}} = 2,5 \cdot 92 = 230 \ A$$

$U_{\text{зт max}}$ – максимальної напруги струму (вихідна напруга швидкості), приймемо $U_{\text{зт max}} = 24 \ B$

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

$$k_{\text{т}} = \frac{24}{230} = 0,104 \ B / A .$$

Передавальна функція замкнутого контура струму, оптимізованого за модульним критерієм, має :

$$W_{\text{т}}(p) = \frac{1/k_{\text{т}}}{2T_{\mu\text{т}}^2 p^2 + 2T_{\mu\text{т}} p + 1} . \qquad (2.4)$$

2.8 Синтез послідовного корегуючого пристрою контура ЕРС

На рис. 2.4 наведена алгоритмічна схема системи управління із контуром струму, оптимізованим за модульним критерієм. Оскільки доданком $2T_{\mu\text{т}}^2 p^2$ у знаменнику передавальної функції (2.4) зазвичай нехтують, то йконтур струму представлений передавальною функцією:

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

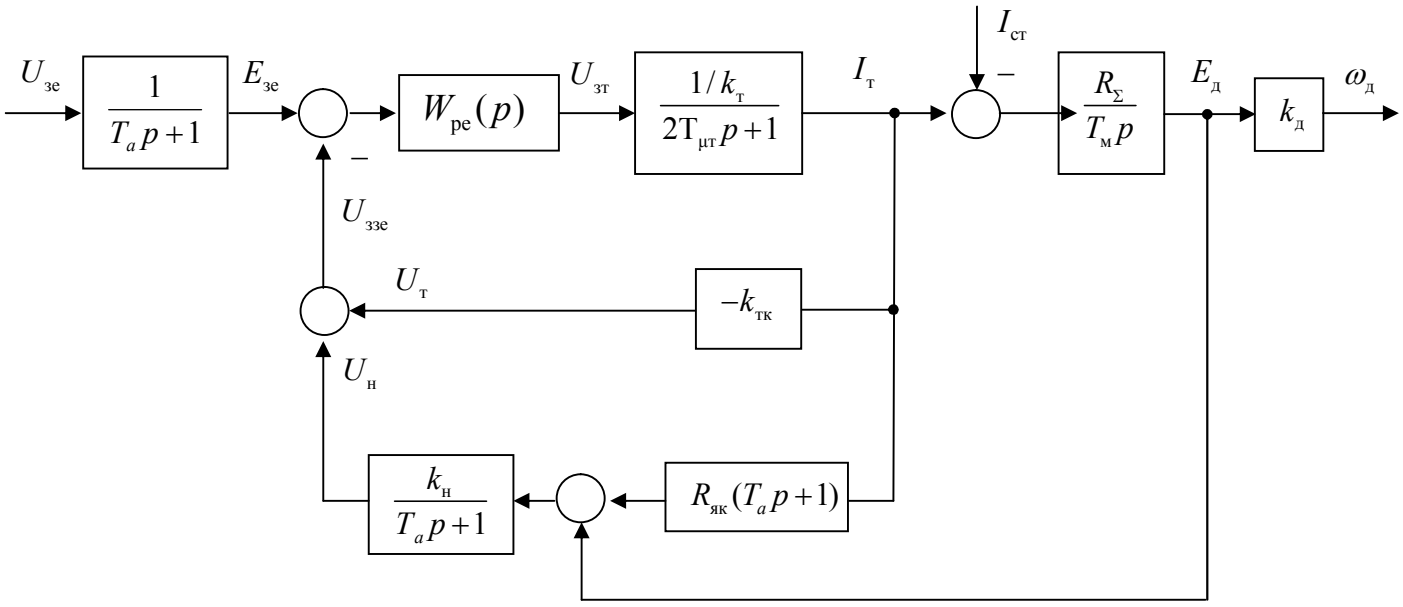


Рисунок 2.4 - Блок-схема алгоритму управління з використанням зворотного зв’язку по ЕРС

Для побудови коректуючого пристрою перетворюємо схему до вигляду на рис. 2.5, за умови, що коефіцієнт посилення ланцюга струмової компенсації визначений як:

$$k_{тк} = k_н R_{як} \ ,$$

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

Аперіодична ланка $\frac{1}{T_a p+1}$ введена на вхід системи для зменшення перерегулювання при зміні задавального сигналу. Контур ЕРС оптимізується за симетричним критерієм. Для побудови послідовного коригуючого пристрою контуру ЕРС схему, що наведена на рисунку 2.5, перетворюємо на еквівалентну систему з одиничним зворотним зв’язком. Розрахункова схема такої системи представлена на рисунку 2.6.

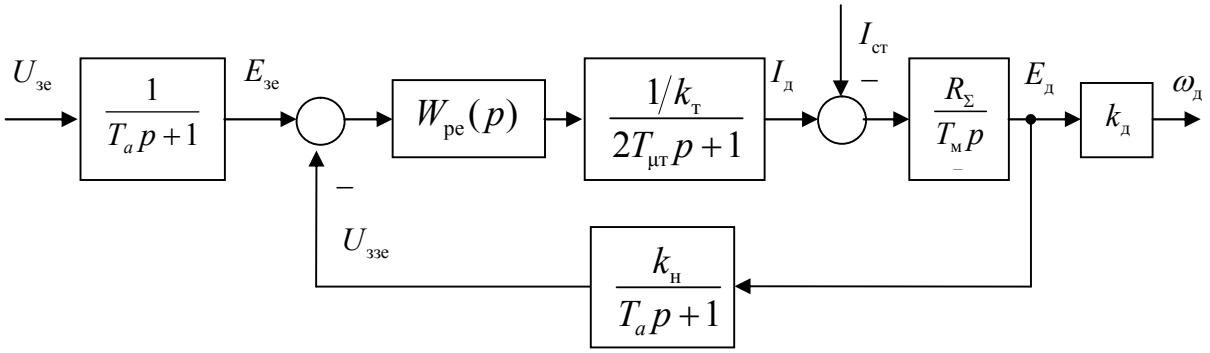


Рисунок 2.5 – Перетворена блок-схема системи управління

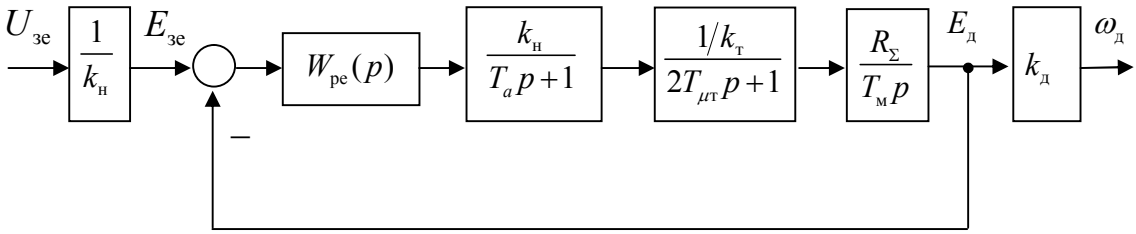


Рисунок 2.6 – Розрахункова схема системи
з одиничним зворотним зв’язком

Як слідує з рис. 2.6, передавальна функція об'єкту регулювання контура ЕРС визначається як:

$$W_{об\,е} = \frac{k_н R_Σ / k_т}{pT_м(T_{μе}p+1)},$$

де $T_{μе}$ – мала постійна часу контура ЕРС, що не компенсується.

$$T_{μе} = T_a + 2T_{μт},$$

$$T_{μе} = 0,017 + 2 \cdot 0,0026 = 0,043\ c.$$

Через те, що мала некомпенсована стала контуру ЕРС включає T_a, швидкість реакції та статична точність регулювання проектованої системи виявля-

ються у $\frac{T_{\mu e}}{T_{\mu т}}$ разів нижчими порівняно із системою, яка використовує зворотний зв'язок за швидкістю.

Бажаний вигляд передавальної функції розімкненого контуру ЕРС, оптимізованого за симетричним критерієм, можна подати у формі:

$$W_{бе}(p) = \frac{1 + 4T_{\mu e} p}{8T_{\mu e}^2 p^2 (1 + T_{\mu e} p)}.$$

Відповідно, передавальна функція регулятора ЕРС набуває вигляду:

$$W_{ре}(p) = \frac{W_{бе}}{W_{об\,е}(p)} = \frac{\frac{1 + 4T_{\mu e} p}{8T_{\mu e}^2 p^2 (1 + T_{\mu e} p)}}{\frac{k_н R_\Sigma / k_т}{p T_м (T_{\mu e} p + 1)}} = \frac{T_м k_т}{2T_{\mu e} k_н R_\Sigma} \cdot \frac{1 + 4T_{\mu e} p}{4T_{\mu e} p}.$$

Таким чином, отримана передавальна функція відповідає типовій конфігурації ПІ-регулятора..

Визначаємо параметри. що входять до складу контура ЕРС.

Коефіцієнт підсилення зворотного зв'язку по напрузі визнач як відношення максимального значення ЕРС двигуна $E_{д\ max}$ до напруги завдання $U_{зе\ max}$, яка відповідає $E_{д\ max}$:

$$k_н = \frac{U_{зе\ max}}{E_{д\ max}},$$

Приймаємо $U_{зе\ max} = 24\ B$; $E_{д\ max} = U_н = 440B$

$$k_н = \frac{24}{440} = 0,05454.$$

Передавальний коефіцієнт струмової компенсації визначається як:

$$k_{\text{тк}} = k_{\text{н}} \cdot R_{\text{як}},$$

$$k_{\text{тк}} = 0,05454 \cdot 0,136 = 0,0074.$$

Алгоритмічна схема системи представлена на рис. 2.7.

Передавальні функції системи можна подати у вигляді:

$$W_{\omega_{\text{д}}, U_{\text{зе}}}(p) = \frac{\omega_{\text{д}}(p)}{U_{\text{зе}}(p)} = \frac{k_{\text{д}}}{k_{\text{н}}} \cdot \frac{B(p)}{A(p)};$$

$$W_{\omega_{\text{д}}, I_{\text{ст}}}(p) = \frac{\omega_{\text{д}}(p)}{I_{\text{ст}}(p)} = -\frac{8T_{\mu\text{е}}^2 k_{\text{д}} R_{\Sigma}}{T_{\text{м}}} \cdot \frac{p(1+T_{\mu\text{е}}p)}{A(p)};$$

$$W_{I_{\text{д}}, U_{\text{зе}}}(p) = \frac{I_{\text{д}}(p)}{U_{\text{зе}}(p)} = \frac{T_{\text{м}}}{k_{\text{н}} R_{\Sigma}} \cdot \frac{pB(p)}{A(p)};$$

$$W_{I_{\text{д}}, I_{\text{ст}}}(p) = \frac{I_{\text{д}}(p)}{I_{\text{ст}}(p)} = \frac{B(p)}{A(p)},$$

де поліноми $A(p)$ і $B(p)$ визначаються наступним чином:

$$A(p) = 8T^3{}_{\mu\text{е}}p^3 + 8T_{\mu\text{е}}^2 p^2 + 4T_{\mu\text{е}}p + 1;$$

$$B(p) = 4T_{\mu\text{е}}p + 1.$$

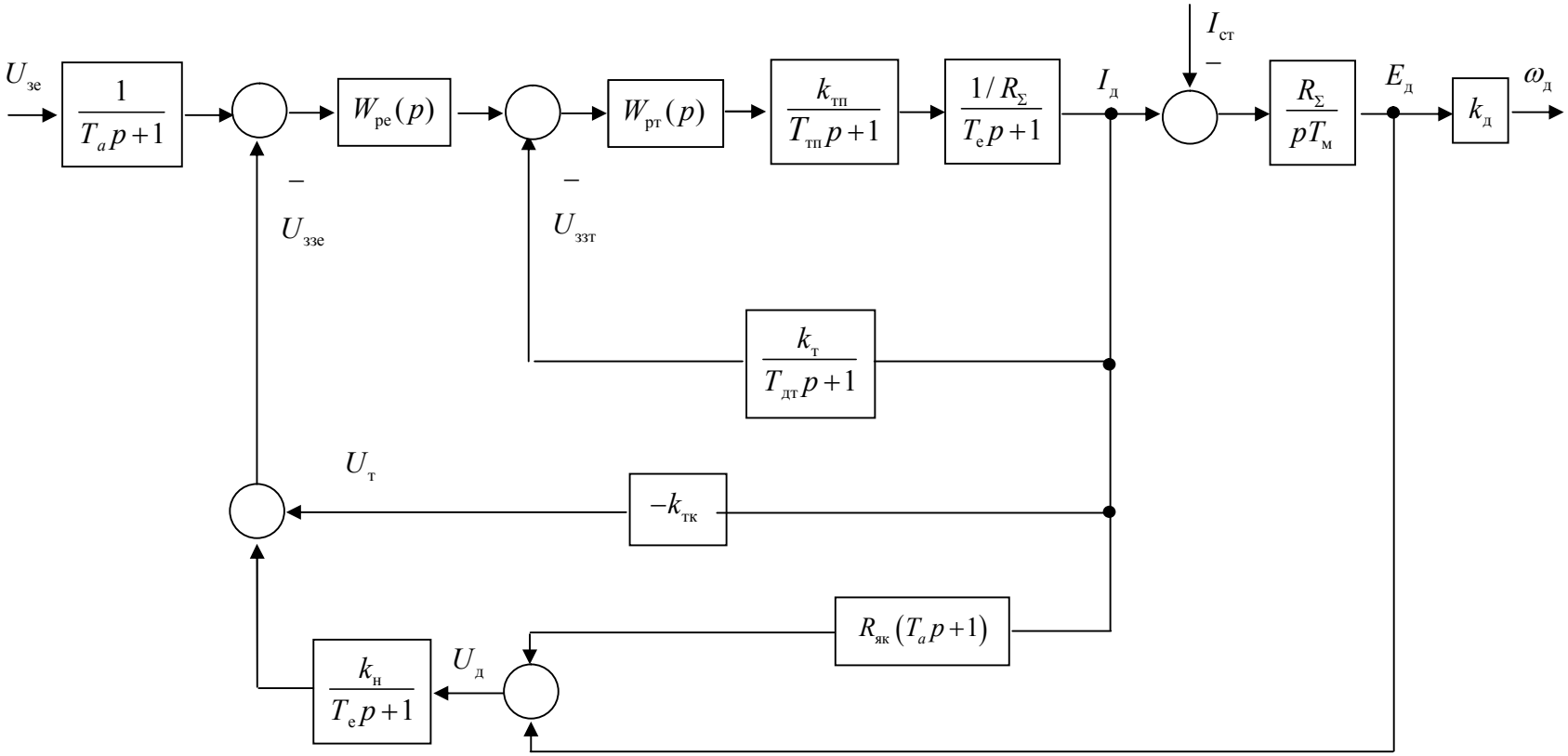


Рисунок 2.7 – Алгоритмічна схема системи підлеглого регулювання швидкості із зворотним зв’язком по ЕРС двигуна

Перетворимо алгоритмічну схему системи підпорядкованого регулювання швидкості, що включає контур струму, налаштований за модульним критерієм, та контур ЕРС, оптимізований за симетричним критерієм, показану на рисунку 2.7, у форму, представлену на рисунку 2.8. Таке перетворення здійснене за умови, що коефіцієнт підсилення ланцюга струмової компенсації прийнятий рівним $k_{\text{тк}} = k_{\text{н}} R_{\text{як}}$, де $k_{\text{н}}$ – коефіцієнт підсилення ланцюга зворотного зв'язку по напрузі.

На схемі, представленій на рисунку 2.8, використані наступні позначення:

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

$k_{\text{пе}}$, k_{ie} – коефіцієнти підсилення пропорційної та інтегральної частин регулятора ЕРС. Значення цих параметрів можуть бути отримані із передавальної функції регулятора ЕРС:

$$W_{\text{ре}}(p) = \frac{T_{\text{м}} k_{\text{с}}}{2T_{\mu\text{е}} k_{\text{н}} R_{\Sigma}} \cdot \frac{4T_{\mu\text{е}} p + 1}{4T_{\mu\text{е}} p} = \frac{T_{\text{м}} k_{\text{т}}}{2T_{\mu\text{е}} k_{\text{н}} R_{\Sigma}} + \frac{T_{\text{м}} k_{\text{с}}}{8T_{\mu\text{е}}^2 k_{\text{н}} R_{\Sigma}} \frac{1}{p} = k_{\text{пе}} + \frac{k_{\text{ie}}}{p},$$

$$k_{\text{пе}} = \frac{T_{\text{м}} k_{\text{с}}}{2T_{\mu\text{е}} k_{\text{н}} R_{\Sigma}}, \quad k_{\text{ie}} = \frac{T_{\text{м}} k_{\text{с}}}{8T_{\mu\text{е}}^2 k_{\text{н}} R_{\Sigma}}.$$

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

$k_{\text{пе}}$, k_{ie} – коефіцієнти підсилення пропорційної та інтегральної частин регулятора струму. Значення цих параметрів визначаються з передавальної функції регулятора струму:

$$W_{\text{рс}}(p) = \frac{R_{\Sigma}}{2T_{\mu\text{с}} k_{\text{тп}} k_{\text{с}}} \cdot \frac{1 + T_{\text{е}} p}{p} = \frac{R_{\Sigma} T_{\text{е}}}{2T_{\mu\text{с}} k_{\text{тп}} k_{\text{с}}} + \frac{R_{\Sigma}}{2T_{\mu\text{с}} k_{\text{тп}} k_{\text{с}}} \frac{1}{p} = k_{\text{пс}} + \frac{k_{\text{ic}}}{p},$$

$$k_{\text{пс}} = \frac{R_{\Sigma} T_{\text{е}}}{2T_{\mu\text{т}} k_{\text{тп}} k_{\text{с}}}, \quad k_{\text{ic}} = \frac{R_{\Sigma}}{2T_{\mu\text{с}} k_{\text{тп}} k_{\text{с}}}.$$

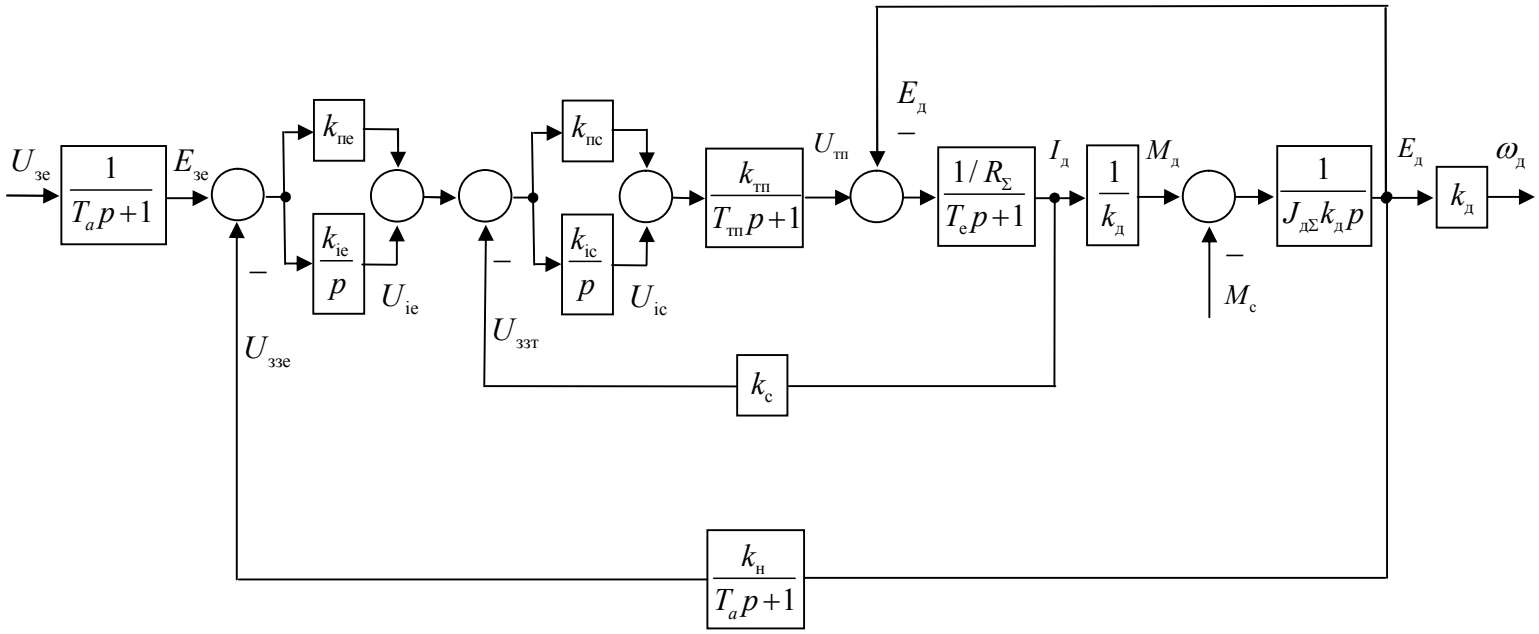


Рисунок 2.8 – Перетворена алгоритмічна схема системи

В передавальній функції двигуна замість електромеханічної постійної часу $T_м$ записано:

$$T_м = \frac{R_\Sigma \cdot J_{д\Sigma}}{\left(k\Phi_н\right)^2} = R_\Sigma J_{д\Sigma} \cdot k_д^2; \quad k_д = {1}/{k\Phi_н}.$$

На схемі також відображенозворотний зв'язок по ЕРС двигуна.

Спрощена версія алгоритмічної схеми показана на рис. 2.9. Контур струму представлено як послідовне з'єднання аперіодичної та інтегруючої ланки з одиничним зворотним зв'язком. Сигнал на виході першої ланки пропорційний ривку електроприводу P.

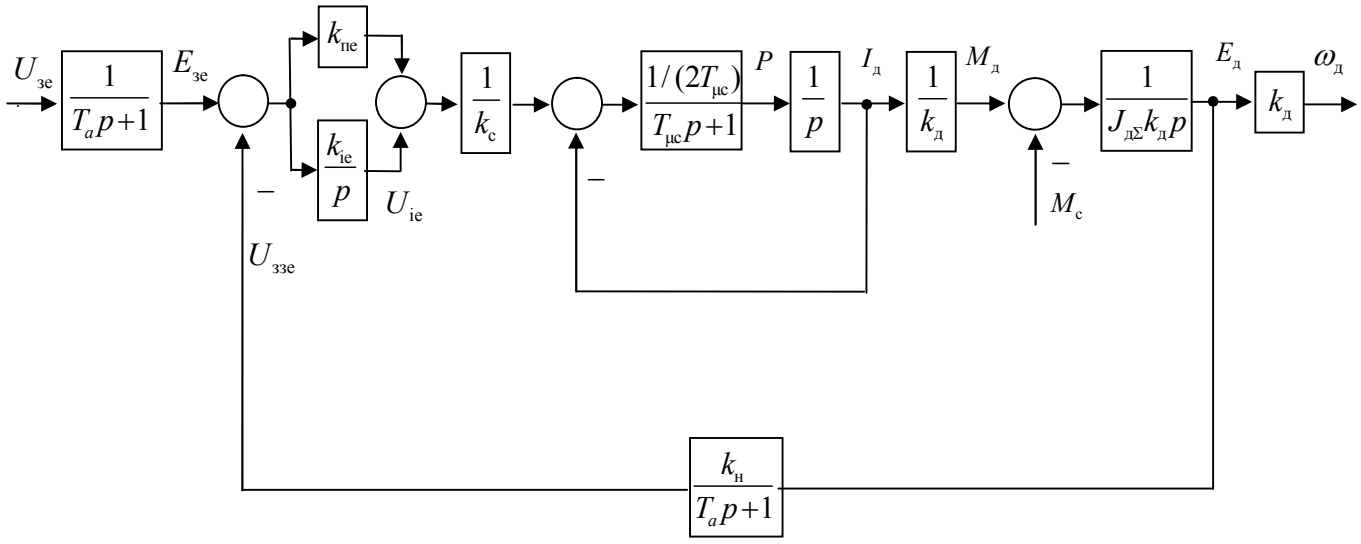


Рисунок 2.9 – Спрощена алгоритмічна схема системи управління

2.9 Дослідження динамічних характеристик системи

Перехідні процеси системи визначались з застосуванням середовища MATLAB. Ця система дозволяє виконувати розрахунки перехідних процесів на основі передавальних функцій системи, векторно-матричних рівнянь або за допомогою схемних моделей, створених у SIMULINK.

Для моделювання системи була розроблена модель, представлена на рисунку 2.10. Схема моделі побудована відповідно до алгоритмічної схеми, наведеної на рисунку 2.8. Спрощена версія моделі системи, створена на основі алгоритмічної схеми рисунку 2.9, наведена на рисунку 2.11.

На рисунках 2.12 і 2.13 показані графіки перехідних процесів швидкості $\omega_{\text{д}}$ та струму двигуна $I_{\text{д}}$ у відповідь на задавальний сигнал. При налаштуванні системи на симетричний оптимум спостерігається суттєве перерегулювання швидкості, яке складає 43,3%.

Для зменшення перерегулювання доцільно включити вхідний фільтр із постійною часу $4T_{\mu\text{e}}$. У результаті перерегулювання знижується до 8,1%, проте при цьому збільшується час першого узгодження системи. Графіки $\omega_{\text{д}} = f(t)$ і $I_{\text{д}} = f(t)$ з фільтром та без нього позначені як «1» і «2» відповідно.

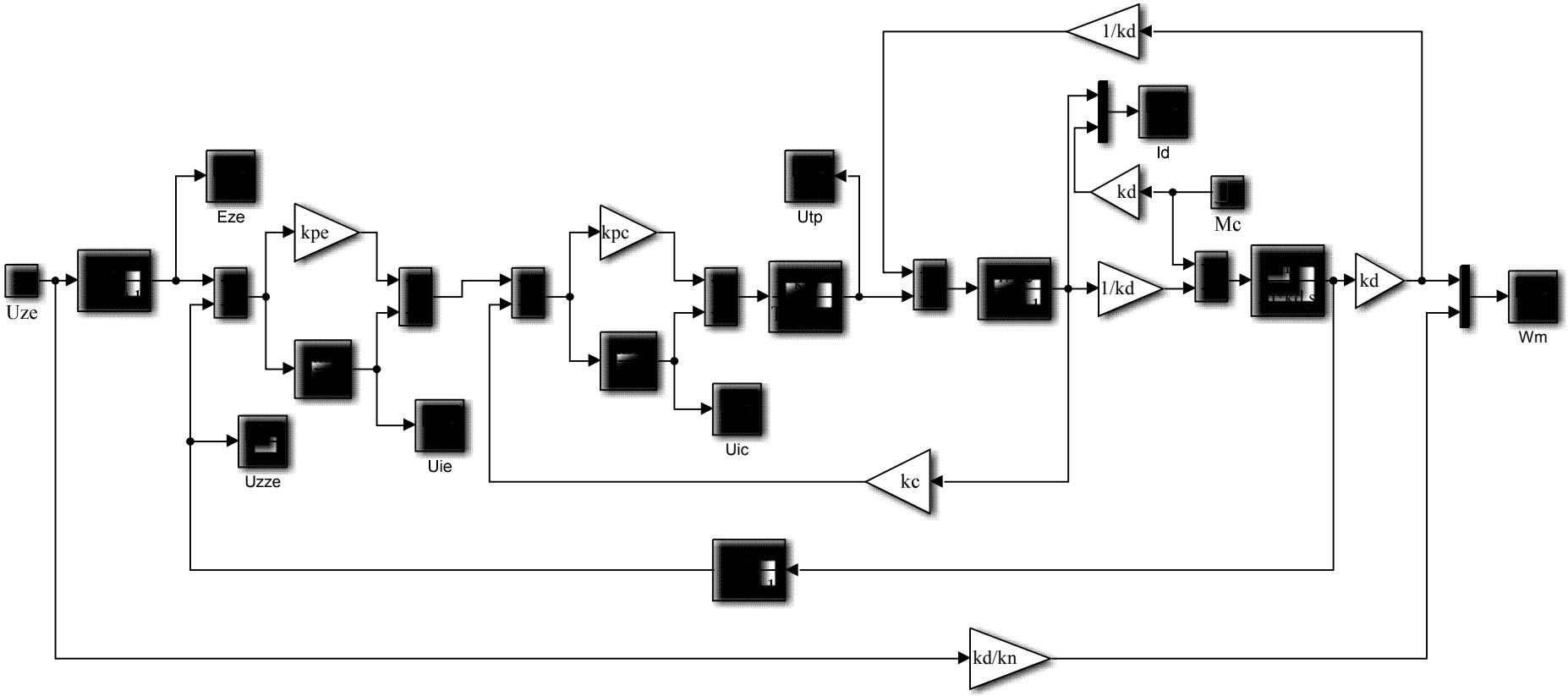


Рисунок 2.10 – Модель системи, реалізована у середовищі MATLAB/SIMULINK

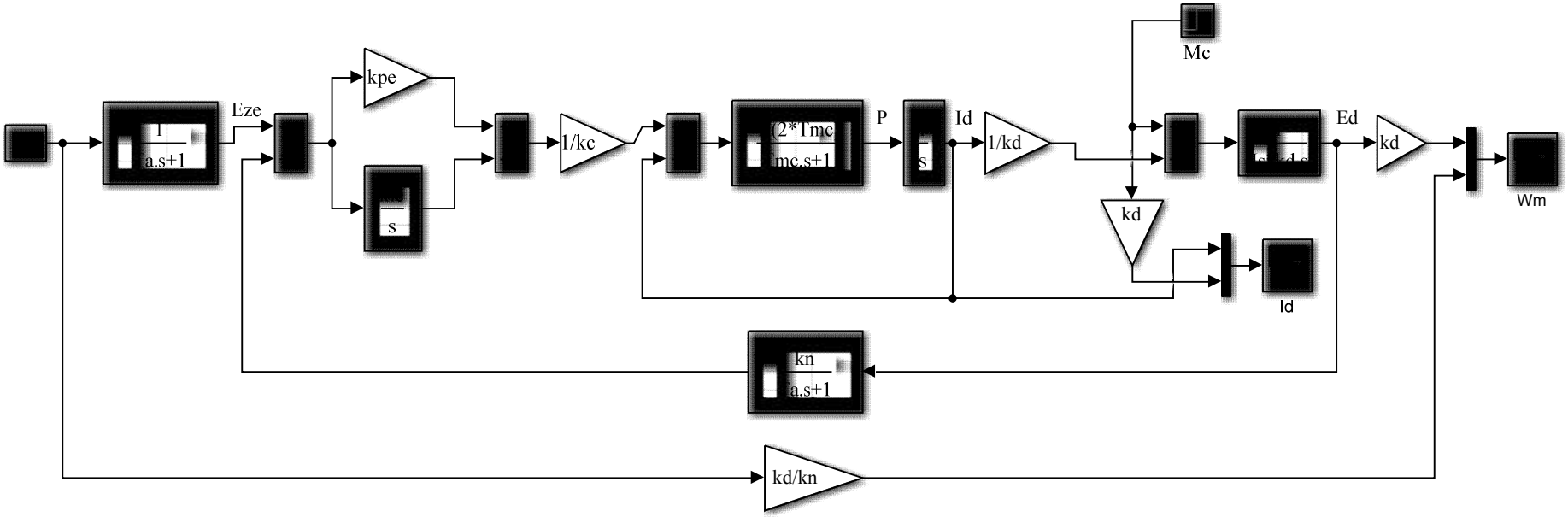
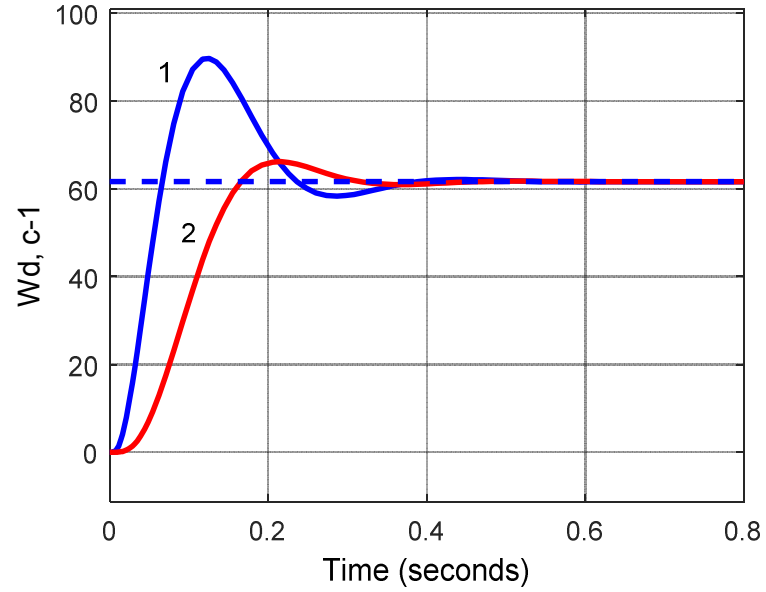
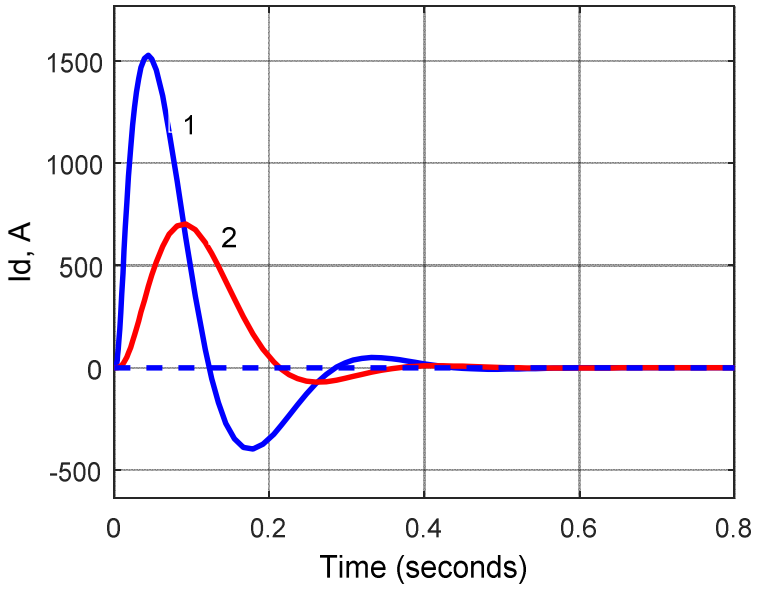


Рисунок 2.11 – Спрощена версія моделі системи у MATLAB/SIMULINK

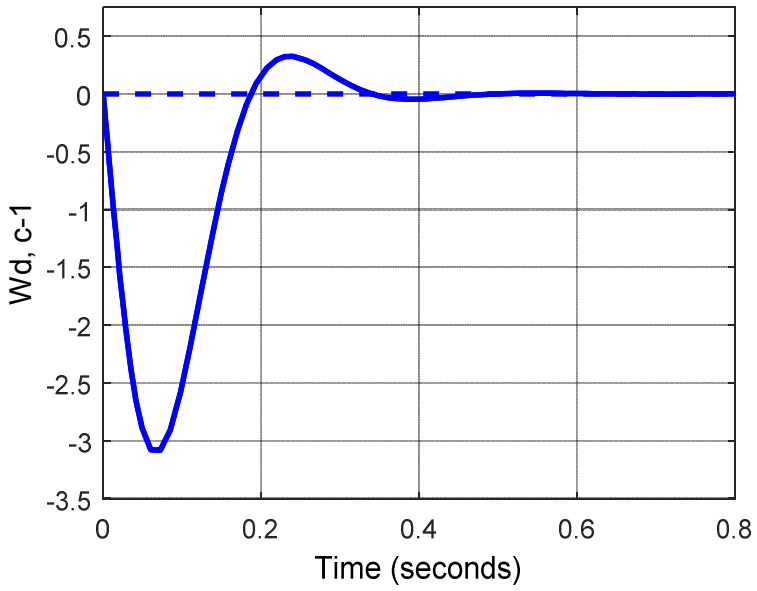


а)

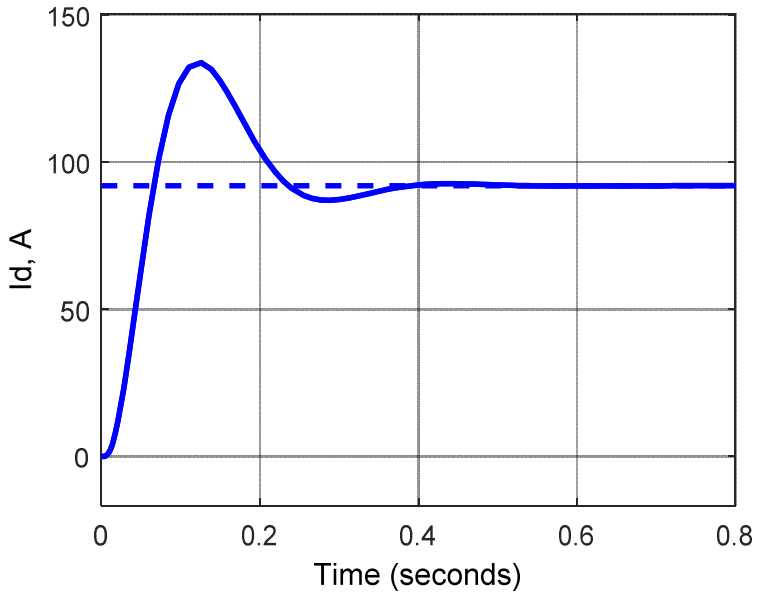


б)

Рисунок 2.12 – Перехідні процеси швидкості $\omega_д = f(t)$ а) і струму $I_д = f(t)$ б) по задаючій дії



а)



б)

Рисунок 2.13 – Перехідні процеи швидкості $\omega_д = f(t)$ а) і струму $I_д = f(t)$ б) по збурюючій дії

ВИСНОВКИ ДО РОЗДІЛУ 2

1. На основі аналізу технічних вимог до систем електроприводів основних механізмів мостових кранів для приводу механізму підйому обрано електропривод постійного струму з конфігурацією ТП-Д. Для забезпечення точного керування координатами приводу застосована система підлеглого регулювання.

2. Проведено вибір електродвигуна та основного силового обладнання електроприводу. Виконана перевірка двигуна за умовами нагріву та перевантажувальної здатності шляхом побудови швидкісних та навантажувальних діаграм. Розрахункове значення еквівалентного моменту показало, що двигун відповідає умовам експлуатації за нагрівом та перевантажувальною здатністю.

3. Розраховано параметри силового кола електроприводу, які необхідні для побудови системи автоматичного управління.

4. Вибрано конфігурацію системи автоматичного управління, яка забезпечує необхідні механічні характеристики електроприводу як у статичних, так і в динамічних режимах, включно з високою якістю перехідних процесів і оптимізацією роботи в різних режимах функціонування механізму. Для електроприводу механізму рольганга застосовано двоконтурну систему підлеглого регулювання: один контур відповідає за струм, інший – за швидкість, при цьому система замкнена по ЕРС двигуна.

5. Виконано синтез послідовних коригуючих пристроїв для окремих контурів. Оскільки характер перехідних процесів залежить від співвідношення постійних часу елементів системи, для забезпечення високоякісних переходів застосовано регулятори, що коригують ці співвідношення. Для контуру напруги синтезовано П-регулятор, а для контурів струму та ЕРС – ПІ-регулятори. В результаті контур струму оптимізовано за модульним критерієм, а контур ЕРС – за симетричним критерієм.

6. Проведено моделювання розробленої системи з використанням пакету MATLAB та середовища SIMULINK. Створена модель системи дозволяє досліджувати перехідні процеси як по задавальних, так і по збурюючих впливах. Аналіз перехідних процесів здійснювався також на основі векторно-матричних рівнянь та передавальних функцій системи. При налаштуванні на симетричний оптимум спостерігалося значне (43,3%) перерегулювання швидкості. Для його зменшення до виходу системи введено фільтр з постійною часу $4T_{\mu e}$, що дозволило знизити перерегулювання до $8{,}1\%$.

РОЗДІЛ 3
ДОСЛІДЖЕННЯ ДИНАМІЧНИХ ХАРАКТЕРИСТИК СИСТЕМИ З УРАХУВАННЯМ КІНЦЕВОЇ ЖОРСТКОСТІ ПІДЙОМНОГО КАНАТА

3.1 Розрахункові схеми механічної частини електроприводу

Побудова системи управління проводилась з початковим припущенням, що всі механічні елементи електроприводу, включаючи підйомний канат, мають практично нескінченну жорсткість, тобто розглядалася одномасова модель. В реальних умовах зв'язки між елементами мають кінцеву жорсткість, а підйомний канат представляє пружний елемент із розподіленими параметрами. Тому механізм підйому мостового крана слід розглядати як багатомасову систему, що включає кілька мас, які рухаються як поступально, так і обертально.

Кінематичну схему електроприводу механізму підйому мостового крана наведено на рис. 3.1, а. Двигун через сполучну муфту $СМ1$, клиноремінну передачу $КРП$ та ряд зубчатих передач $ЗП_1$ ..., $ЗП_i$ і сполучну муфту приводить у рух барабан $Б$, який перетворює обертальний рух на поступальний рух вантажозахватного пристрою. Вантаж $В$ має масу $m_в$, яка переміщується із швидкістю $V_в$ і підпорядковується силі тяжіння $F_в$.

Схема демонструє, що механічна частина електроприводу є системою взаємопов'язаних мас, які рухаються з різними швидкостями. Під дією навантажень елементи системи (вали, опори, клиноремінні та зубчаті передачі, канати) деформуються, оскільки механічні зв'язки не є абсолютно жорсткими. Переміщення мас визначається жорсткістю цих зв'язків.

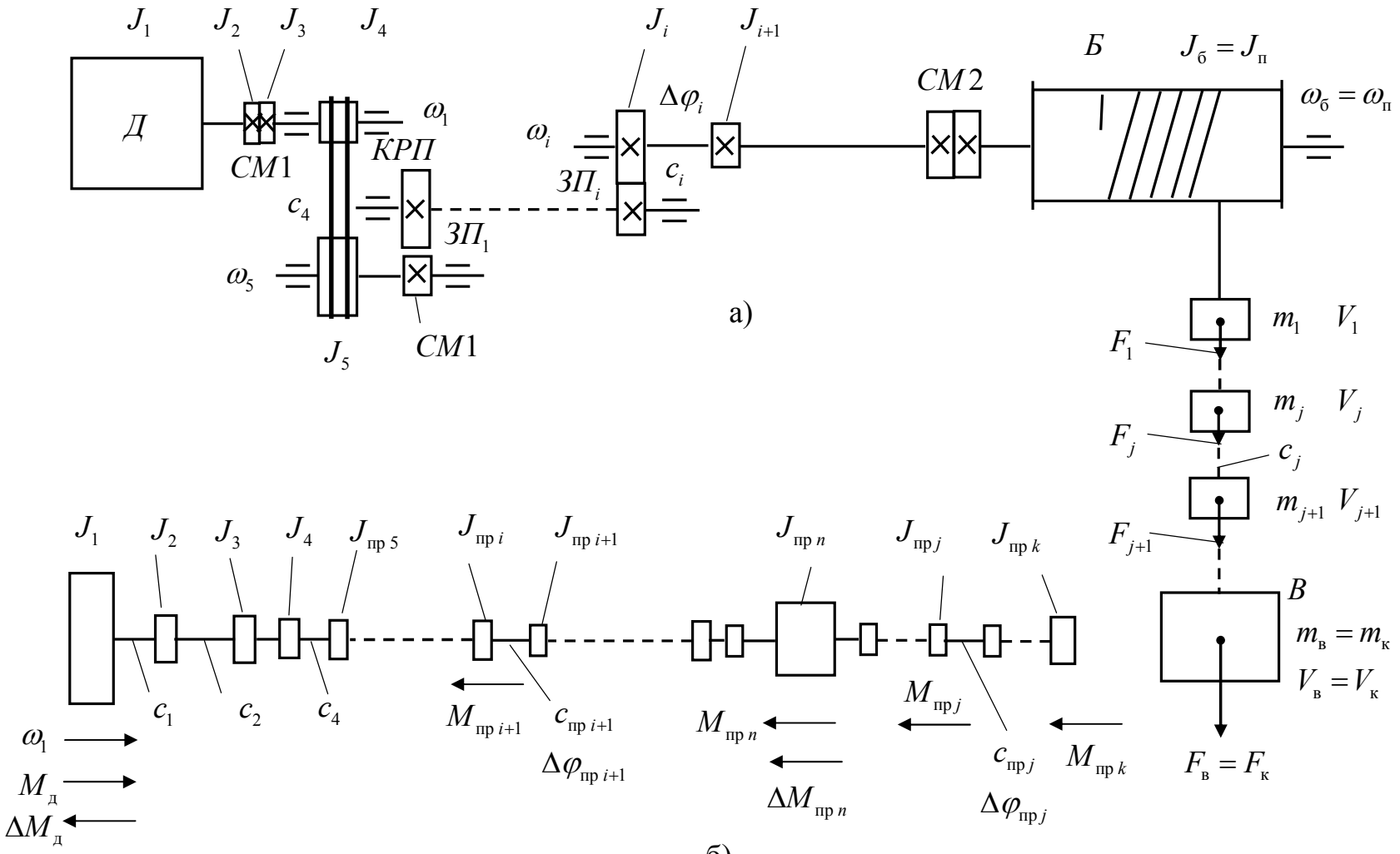


Рисунок 3.1 – Схеми механічної частини електроприводу:

а) кінематична схема; б) розрахункова схема для дослідження динаміки

При складанні кінематичної схеми враховано, що механічна частина приводу містить n обертально рухомих мас і k поступально рухомих мас. Кожен обертально рухомий елемент має момент інерції J_i і з'єднаний із сусіднім $(i+1)$-м елементом через механічний зв'язок із жорсткістю c_i. Поступально рухомі елементи мають масу m_j і з'єднані з наступним елементом жорстким пружним зв'язком з жорсткістю c_j. Жорсткість пружних зв'язків підпорядковується закону Гука і визначається співвідношеннями:

$$c_i = M_{\text{пр}\,i}\Delta\varphi_i, \qquad c_j = F_{\text{пр}\,j}\Delta S_j,$$

де $M_{\text{пр}\,i}$ і $F_{\text{пр}\,j}$ – момент і сила, що діють на пружний елемент, а $\Delta\varphi_i$; і ΔS_j – відповідні обертальні та поступальні деформації.

Маси окремих елементів та жорсткості їхніх з'єднань у кінематичному ланцюзі електроприводу можуть значно відрізнятися. Найбільший вплив на динаміку системи чинять масивні компоненти та з'єднання з найменшою жорсткістю. Тому одним із початкових етапів проектування та аналізу електроприводів є побудова спрощених розрахункових схем механічної частини, що дозволяють ігнорувати пружність жорстких зв'язків і приблизно враховувати вплив малих мас.

Через наявність передач різні вузли системи можуть рухатися з різними швидкостями, тому їхні моменти інерції J_i, маси m_j, жорсткості c_i і c_j, та переміщення $\Delta\varphi_i$; і ΔS_j не можна безпосередньо порівнювати. Щоб зробити порівняння можливим, усі параметри елементів приводять до єдиної розрахункової швидкості – зазвичай до швидкості двигуна, хоча інколи доцільно брати швидкість іншого вузла, наприклад механізму при його поступальному русі.

При приведенні параметрів розрахункова схема повинна відповідати законам збереження енергії: необхідно зберігати кінетичну та потенційну енергії системи, а також елементарну роботу всіх сил і моментів під час можливих переміщень. Формули приведення моменту інерції обертового елемента або маси, що рухається поступально, до розрахункової швидкості мають вигляд:

$$J_{\text{пр}\,i} = J_i / i_{1i}^2, \qquad J_{\text{пр}\,j} = m_j / \rho_{1j}^2,$$

де $i_{1i} = \omega_1 / \omega_i$ – передавальне число від валу, який приводиться, до розрахункового;

$\rho_{1j} = V_j / \omega_1$ – радіус, приведений до розрахункової швидкості ω_1.

Аналогічно, при приведенні обертальних та поступальних переміщень слід враховувати співвідношення швидкостей:

$$\Delta\varphi_{\text{пр}\,i} = \varphi_i i_{1i}, \qquad \varphi_{\text{пр}\,j} = S_j / \rho_{1j}.$$

Жорсткості зв'язків приводять так, щоб зберегти потенційну енергію пружних елементів:

$$c_{\text{пр}\,i} = c_i / i_{1i}^2, \qquad c_{\text{пр}\,j} = c_j / \rho_{1j}^2.$$

Моменти і сили навантаження також потрібно приводити, виходячи з умови рівності елементарної роботи:

$$M_{\text{пр}\,i} = M_i / i_{1i}, \qquad M_{\text{пр}\,j} = F_j \rho_{1j}.$$

На практиці відомі моменти інерції, маси і жорсткості реальних елементів, а сили або задаються, або визначаються з умов руху механізму та технологічних вимог. Після приведення всіх параметрів до єдиної швидкості можна виділити основні маси та пружні з'єднання і побудувати наближену розрахункову схему механічної частини. Для наочності маси часто зображують у вигляді прямокутників, площа яких пропорційна приведеному моменту інерції, а жорсткості – у вигляді з'єднань, довжина яких обернено пропорційна жорсткості.

Наприклад, для кінематичної схеми, показаної на рис. 1.1,а, розрахункова схема (рис. 1.1,б) виділяє три основні маси: ротор двигуна J_1, барабан із приведеним моментом інерції $J_{\text{пр}\,n}$ та вантаж $J_{\text{пр}\,k}$. Менші моменти інерції можна врахувати, додавши їх до найближчих великих мас, а жорсткості зв'язків між масами визначають за формулою еквівалентних жорсткостей. На схемі вказано

також прикладені зовнішні моменти та електромагнітний момент двигуна, враховано напрямок позитивної швидкості. У спрощеній схемі всі зовнішні сили та моменти сумуються, а з'єднання між масами вважаються жорсткими.

Практика показує, що розрахункові схеми без розгалужень після виділення основних мас та жорсткостей часто зводяться до трьохмасових або двомасових систем (рис. 3.2).

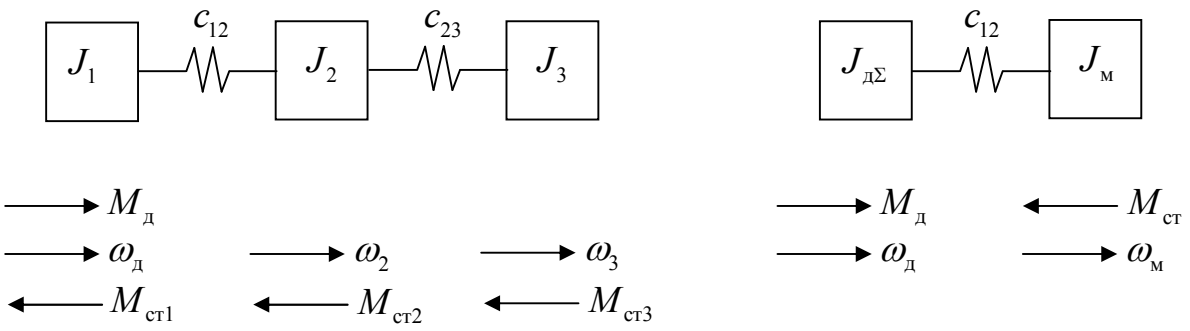


Рисунок 3.2 – Розрахункові схемі механічної частини електроприводу
а) – схема трьохмасової системи; б) – схема двомасової системи

Для трьохмасової системи (рис. 3.2,а) параметрами є приведені моменти інерції мас J_1 ,J_2 і J_3, і еквівалентні жорсткості між ними c_1 і c_2. До першої маси прикладені електромагнітний момент M і момент статичного навантаження $M_{ст1}$; до проміжної – момент опору $M_{ст2}$, а до третьої – момент зовнішнього навантаження $M_{ст3}$. Така схема застосовується для детального аналізу руху мас у системах автоматизованого електроприводу і моделюється на ЕОМ.

Двомасова система (рис. 3.2,б) є спрощеною моделлю, що найбільше підходить для дослідження впливу пружних механічних зв'язків. Сумарний приведений момент елементів, що жорстко з'єднані з двигуном, позначається $J_{дΣ}$, а з робочим органом механізму – $J_м$. Пружний зв'язок між ними має еквівалентну жорсткість c_{12}, а зовнішній момент навантаження позначається $M_{ст}$. Така схе-

ма є основною для дослідження впливу пружних з'єднань на динаміку електроприводу.

3.2 Рівняння стану одномасової системи

Для побудови математичної моделі двомасової системи спочатку розглянемо одномасову модель.

Рівняння стану одномасової системи, що відповідає алгоритмічній схемі, показаній на рис. 2.11, має вигляд:

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

Цю систему можна подати у векторно-матричній формі, доповнивши рівнянням виходу. Введемо вектор стану $\vec{X}(t)$ розмірності n та вектор керування $\vec{U}(t)$ розмірності m:

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

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

де T – означає операцію транспонування.

Ррівняння стану та виходу системи у матричній формі записуються як:

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

де $\mathbf{A}$ – матриця стану системи розміром $n \times n$;

$\mathbf{B}$ – матриця керцвання розміром $n \times m$;

$\mathbf{C}$ - матриця виходу розміром $r \times n$;

$\vec{Y}(t)$ - вектор виходу розміром r.

Матриці $\mathbf{A}$ і $\mathbf{B}$ одномасової системи мають наступну структуру:

$$\mathbf{A} = \begin{array}{c} \begin{matrix} \omega_д & I_д & U_{тп} & U_{ic} & U_{ie} & E_{зе} & U_{ззе} \end{matrix} \\ \begin{vmatrix} 0 & \frac{k\Phi_н}{J_{дΣ}} & 0 & 0 & 0 & 0 & 0 \\ -\frac{k\Phi_н}{R_Σ T_e} & -\frac{1}{T_e} & \frac{1}{R_Σ T_e} & 0 & 0 & 0 & 0 \\ 0 & -\frac{k_c k_{пт} k_{тп}}{T_{μc}} & -\frac{1}{T_{μc}} & \frac{k_{тп}}{T_{μc}} & \frac{k_{тп}k_{пс}}{T_{μc}} & \frac{k_{пе}k_{пс}k_{тп}}{T_{μc}} & -\frac{k_{пе}k_{пс}k_{тп}}{T_{μc}} \\ 0 & -k_{ic}k & 0 & 0 & k_{ic} & k_{пе}k_{ic} & -k_{пе}k_{ic} \\ 0 & 0 & 0 & 0 & 0 & k_{ie} & -k_{ie} \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_a} & 0 \\ \frac{k_н}{k_д T_a} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_a} \end{vmatrix} \end{array} \quad \mathbf{B} = \begin{array}{c} \begin{matrix} U_{зе} & M_{ст} \end{matrix} \\ \begin{vmatrix} 0 & -\frac{1}{J_{дΣ}} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{T_a} & 0 \\ 0 & 0 \end{vmatrix} \end{array}$$

Для спрощеної схеми одномасової системи оптимізований по модульному критерію контур представлений як послідовне з'єднання аперіодичної та інтег-

руючої ланки, охоплених одиничним зворотним зв'язком. Сигнал на виході першої ланки пропорційний ривку електроприводу P.

Рівняння стану спрощеної схеми мають вид:

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

При запису рівнянь стану спрощеної системи у векторно-матричній формі вектор стану $\vec{X}(t)$ та керування $\vec{U}(t)$ набувають виду:

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

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

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

$$\mathbf{A}=\begin{array}{c}\begin{matrix}\omega_{\text{д}} & M_{\text{д}} & P & E_{\text{зе}} & U_{\text{ззе}} & U_{\text{іе}}\end{matrix}\\ \begin{vmatrix} 0 & \frac{1}{J_{\text{д}\Sigma}} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{k_{\text{д}}} & 0 & 0 & 0 \\ 0 & -\frac{k_{\text{д}}}{2T_{\text{µс}}^2} & -\frac{1}{T_{\text{µс}}} & \frac{k_{\text{ре}}}{2T_{\text{µс}}^2 k_{\text{с}}} & -\frac{k_{\text{ре}}}{2T_{\text{µс}}^2 k_{\text{с}}} & \frac{1}{2T_{\text{µс}}^2 k_{\text{с}}} \\ 0 & 0 & 0 & -\frac{1}{T_a} & \frac{1}{T_a} & 0 \\ \frac{k_{\text{н}}}{T_a k_{\text{д}}} & 0 & 0 & 0 & -\frac{1}{T_a} & 0 \\ 0 & 0 & 0 & k_{\text{іе}} & -k_{\text{іе}} & 0 \end{vmatrix}\end{array} \qquad \mathbf{B}=\begin{array}{c}\begin{matrix}U_{\text{зе}} & M_{\text{ст}}\end{matrix}\\ \begin{vmatrix} 0 & -\frac{1}{J_{\text{д}\Sigma}} \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{T_a} & 0 \\ 0 & 0 \\ 0 & 0 \end{vmatrix}\end{array}$$

3.3 Математична модель двомасової системи

Кінематична схема механічної частини електроприводу механізму підйому, зведена до двомасової моделі, у загальному випадку з урахуванням моменту внутрішнього в'язкого тертя описується рівняннями:

$$\begin{cases} J_{\text{м}}\dfrac{d\omega_{\text{м}}}{dt}=M_{\Sigma}-M_{\text{ст}}; \\ \dfrac{dM_{\text{пр}}}{dt}=c_{12}(\omega_{\text{д}}-\omega_{\text{м}}); \\ J_{\text{д}\Sigma}\dfrac{d\omega_{\text{д}}}{dt}=M_{\text{д}}-M_{\Sigma}. \end{cases} \tag{3.2}$$

де M_{Σ} – сумарний момент, що передається через пружне з'єднання, який складається з пружного моменту $M_{\text{пр}}$ та моменту внутрішнього в'язкого тертя $M_{\text{вт}}=\beta_{12}(\omega_{\text{д}}-\omega_{\text{м}})$:

$$M_{\Sigma}=M_{\text{пр}}+\beta_{12}(\omega_{\text{д}}-\omega_{\text{м}}), \tag{3.3}$$

β_{12} – коефіцієнт внутрішнього в'язкого тертя; для розрахунків прийнято $\beta_{12} = 0$.

Підставимо вираз для сумарного моменту (3.3) у (3.2) і запишемо систему диференціальних рівнянь у формі Коші:

$$\begin{cases} \dfrac{d\omega_{\text{м}}}{dt} = \dfrac{1}{J_{\text{м}}} M_{\text{пр}} + \dfrac{\beta_{12}}{J_{\text{м}}} \omega_{\text{д}} - \dfrac{\beta_{12}}{J_{\text{м}}} \omega_{\text{м}} - \dfrac{1}{J_{\text{м}}} M_{\text{ст}}; \\ \dfrac{dM_{\text{пр}}}{dt} = c_{12}\omega_{\text{д}} - c_{12}\omega_{\text{м}}; \\ \dfrac{d\omega_{\text{д}}}{dt} = \dfrac{1}{J_{\text{д}\Sigma}} M_{\text{д}} - \dfrac{1}{J_{\text{д}\Sigma}} M_{\text{пр}} - \dfrac{\beta_{12}}{J_{\text{д}\Sigma}} \omega_{\text{д}} + \dfrac{\beta_{12}}{J_{\text{д}\Sigma}} \omega_{\text{м}}. \end{cases} \tag{3.4}$$

Розглядаємо двомасову систему підлеглого регулювання швидкості, де контур струму налаштований за модульним критерієм, а контур ЕРС – за симетричним критерієм. Алгоритмічна схема такої двомасової системи наведена на рисунку 3.3.

Подібно до одномасової схеми, алгоритмічну модель двомасової системи можна представити у спрощеному вигляді, як показано на рисунку 3.4.

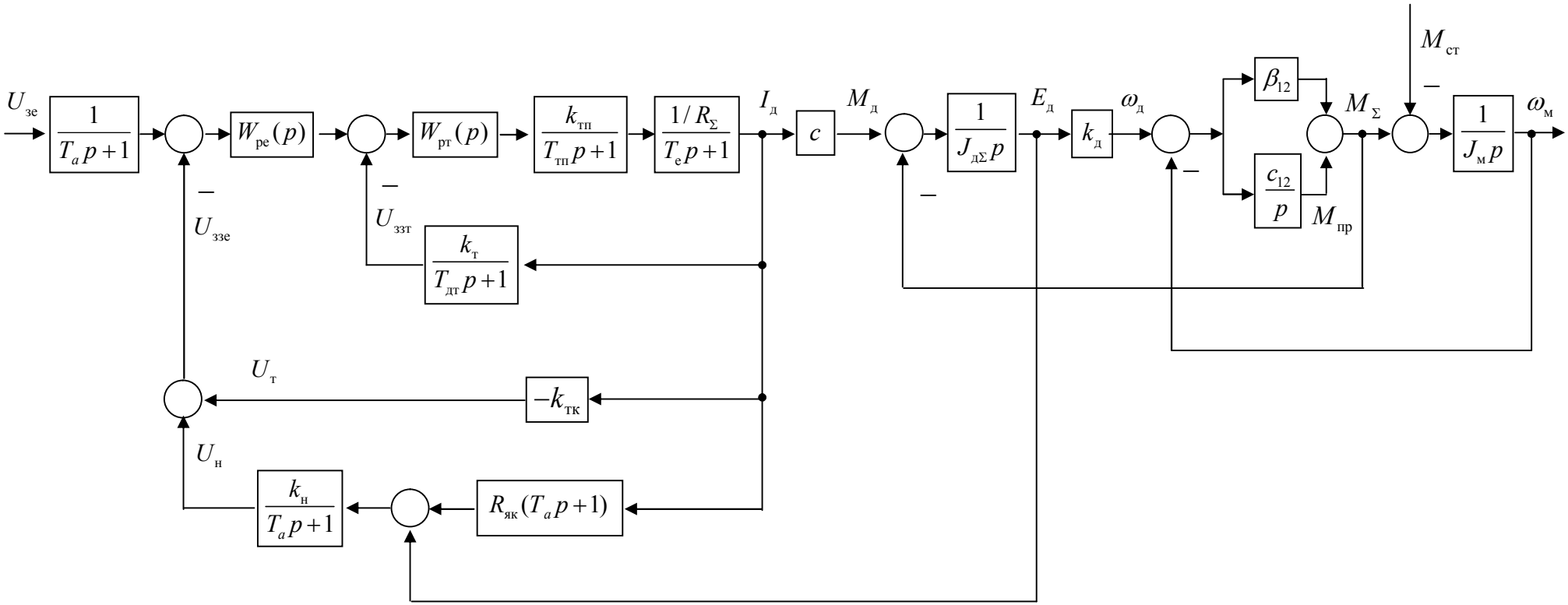


Рисунок 3.3 – Структурна схема двомасової системи підлеглого регулювання

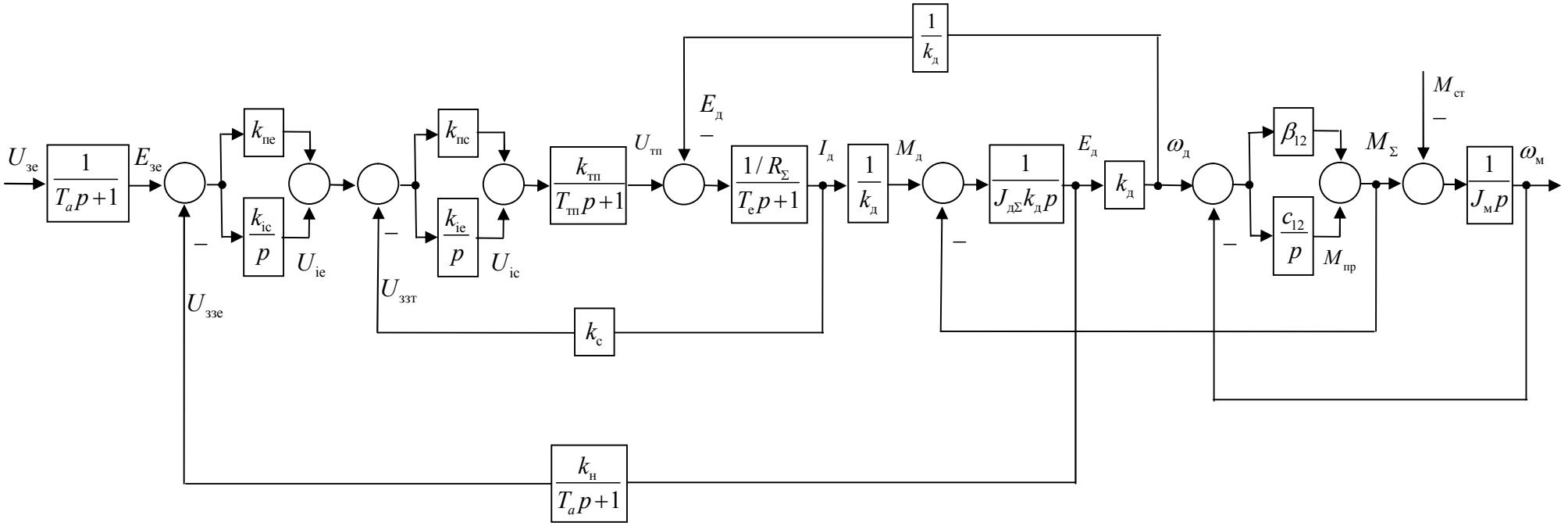


Рисунок 3.4 – Перетворена структурна схема двомасової системи для моделювання

Одномасова система підлеглого регулювання швидкості, у якій контур струму налаштований за модульним критерієм, а контур ЕРС – за симетричним, описується системою диференціальних рівнянь (3.1).

Об’єднавши рівняння систем (3.4) та (3.1), можна отримати систему станів двомасової системи підлеглого регулювання швидкості з зворотним зв’язком по ЕРС двигуна:

$$
\begin{cases}
\dfrac{d\omega_{\text{м}}}{dt} = -\dfrac{\beta_{12}}{J_{\text{м}}}\omega_{\text{м}} + \dfrac{1}{J_{\text{м}}}M_{\text{пр}} + \dfrac{\beta_{12}}{J_{\text{м}}}\omega_{\text{д}} - \dfrac{1}{J_{\text{м}}}M_{\text{ст}}; \\
\dfrac{dM_{\text{пр}}}{dt} = c_{12}\omega_{\text{д}} - c_{12}\omega_{\text{м}}; \\
\dfrac{d\omega_{\text{д}}}{dt} = \dfrac{k\Phi_{\text{н}}}{J_{\text{д}\Sigma}}I_{\text{д}} - \dfrac{1}{J_{\text{д}\Sigma}}M_{\text{пр}} - \dfrac{\beta_{12}}{J_{\text{д}\Sigma}}\omega_{\text{д}} + \dfrac{\beta_{12}}{J_{\text{д}\Sigma}}\omega_{\text{м}}; \\
\dfrac{dI_{\text{д}}}{dt} = -\dfrac{k\Phi_{\text{н}}}{R_{\Sigma}T_{\text{е}}}\omega_{\text{д}} - \dfrac{1}{T_{\text{е}}}I_{\text{д}} + \dfrac{1}{R_{\Sigma}T_{\text{е}}}U_{\text{тп}}; \\
\dfrac{dU_{\text{тп}}}{dt} = -\dfrac{k_{\text{с}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{с}}}I_{\text{д}} - \dfrac{1}{T_{\mu\text{с}}}U_{\text{тп}} - \\
\qquad + \dfrac{k_{\text{тп}}}{T_{\mu\text{с}}}U_{\text{іс}} + \dfrac{k_{\text{тп}}k_{\text{пс}}}{T_{\mu\text{с}}}U_{\text{іе}} + \dfrac{k_{\text{пе}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{с}}}E_{\text{зе}} - \dfrac{k_{\text{пе}}k_{\text{пс}}k_{\text{тп}}}{T_{\mu\text{с}}}U_{\text{ззе}}; \\
\dfrac{dU_{\text{іс}}}{dt} = -k_{\text{іс}}k_{\text{с}}I_{\text{д}} + k_{\text{іс}}U_{\text{іе}} + k_{\text{пе}}k_{\text{іс}}E_{\text{зе}} - k_{\text{пе}}k_{\text{іс}}U_{\text{ззе}}; \\
\dfrac{dU_{\text{іе}}}{dt} = k_{\text{іе}}E_{\text{зе}} - k_{\text{іе}}U_{\text{ззе}}; \\
\dfrac{dE_{\text{зе}}}{dt} = -\dfrac{1}{T_a}E_{\text{зе}} + \dfrac{1}{T_a}U_{\text{зе}}; \\
\dfrac{dU_{\text{ззе}}}{dt} = \dfrac{k_{\text{н}}}{k_{\text{д}}T_a}\omega_{\text{д}} - \dfrac{1}{T_a}U_{\text{ззе}}.
\end{cases}
\tag{3.5}
$$

Систему рівнянь стану двомасової системи зручно записати у векторно-матричній формі, доповнивши її рівнянням виходу. Для цього вводимо вектор стану $\vec{X}(t)$ та вектор управління $\vec{U}(t)$:

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

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

Рівняння стану системи і рівняння виходу у векторно-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 $\mathbf{A}$ – матриця стану двомасової системи;

$\mathbf{B}$ – матриця управління двомасової системи;

$\mathbf{C}$ – матриця виходу двомасової системи;

$\vec{Y}(t)$ – вектор виходу двомасової системи.

Матриці стану $\mathbf{A}$ та управління $\mathbf{B}$ для двомасової системи мають наступну структуру:

Матриці **A** і **B** двомасової системи

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

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

Подібно до того, як алгоритмічна схема одномасової системи може бути спрощена, алгоритмічну модель двомасової системи також можна представити у спрощеній формі, наведеної на рисунку 3.5. У цій системі оптимізований за модульним критерієм контур представлений у вигляді послідовного з'єднання аперіодичної і інтегруючої ланки, охоплених одиничним зворотним зв'язком

В такому випадку рівняння стану двомасової системи записуються у вигляді:

$$\begin{cases} \dfrac{d\omega_{\text{м}}}{dt} = \dfrac{1}{J_{\text{м}}} M_{\text{пр}} + \dfrac{\beta_{12}}{J_{\text{м}}} \omega_{\text{д}} - \dfrac{\beta_{12}}{J_{\text{м}}} \omega_{\text{м}} - \dfrac{1}{J_{\text{м}}} M_{\text{ст}}; \\ \dfrac{dM_{\text{пр}}}{dt} = c_{12}\omega_{\text{д}} - c_{12}\omega_{\text{м}}; \\ \dfrac{d\omega_{\text{д}}}{dt} = \dfrac{1}{J_{\text{д}\Sigma}} M_{\text{д}} - \dfrac{1}{J_{\text{д}\Sigma}} M_{\text{пр}} - \dfrac{\beta}{J_{\text{д}\Sigma}} \omega_{\text{д}} - \dfrac{\beta}{J_{\text{д}\Sigma}} \omega_{\text{м}}; \\ \dfrac{dM_{\text{д}}}{dt} = \dfrac{1}{k_{\text{д}}} P; \\ \dfrac{dP}{dt} = \dfrac{k_{\text{ре}}}{2T_{\mu\text{с}}^2 k_{\text{с}}} E_{\text{зе}} - \dfrac{k_{\text{ре}}}{2T_{\mu\text{с}}^2 k_{\text{с}}} U_{\text{ззе}} - \dfrac{k_{\text{д}}}{2T_{\mu\text{с}}^2} M_{\text{д}} - \dfrac{1}{T_{\mu\text{с}}} P + \dfrac{1}{T_{\mu\text{с}}^2 k_{\text{с}}} U_{\text{іе}}; \\ \dfrac{dE_{\text{зе}}}{dt} = \dfrac{1}{T_a} U_{\text{зе}} - \dfrac{1}{T_a} E_{\text{зе}}; \\ \dfrac{dU_{\text{ззе}}}{dt} = \dfrac{k_{\text{н}}}{T_a k_{\text{д}}} \omega_{\text{д}} - \dfrac{1}{T_a} U_{\text{ззе}}; \\ \dfrac{dU_{\text{іе}}}{dt} = k_{\text{іе}} E_{\text{зе}} - k_{\text{іе}} U_{\text{ззе}}. \end{cases} \tag{3.6}$$

Запишемо рівняння стану (3.6) у векторно-матричній формі.

Вектор стану системи $\vec{X}(t)$ і вектор управління $\vec{U}(t)$ мають вид:

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

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

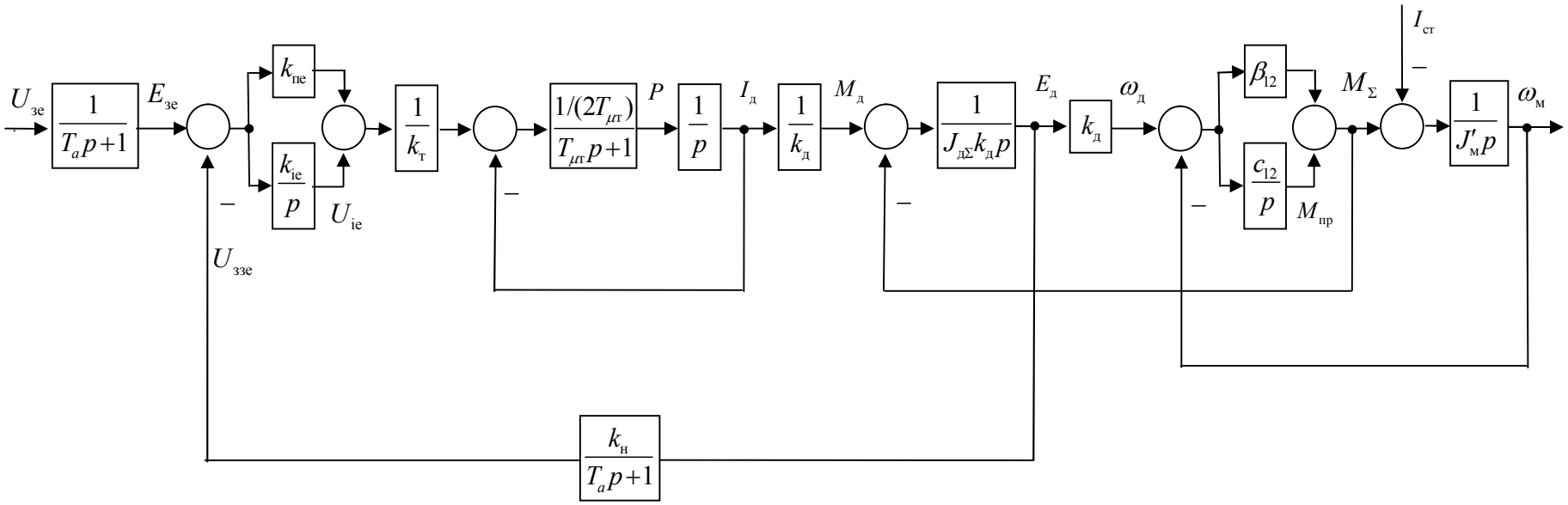


Рисунок 3.5 – Спрощена структурна схема двомасової систем

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

$$\mathbf{A}=\begin{array}{c}\begin{matrix}\omega_{\text{м}} & M_{\text{пр}} & \omega_{\text{д}} & M_{\text{д}} & P & E_{\text{зе}} & U_{\text{ззе}} & U_{\text{іе}}\end{matrix}\\ \begin{vmatrix} -\frac{\beta}{J_{\text{м}}} & \frac{1}{J_{\text{м}}} & \frac{\beta}{J_{\text{м}}} & 0 & 0 & 0 & 0 & 0 \\ -c & & c & 0 & 0 & 0 & 0 & 0 \\ \frac{\beta}{J_{\text{д}}} & -\frac{1}{J_{\text{д}}} & -\frac{\beta}{J_{\text{д}}} & \frac{1}{J_{\text{д}}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{k_{\text{д}}} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{k_{\text{д}}}{2T_{\text{µт}}^2} & -\frac{1}{T_{\text{µт}}} & \frac{k_{\text{ре}}}{2T_{\text{µт}}^2 k_{\text{т}}} & -\frac{k_{\text{ре}}}{2T_{\text{µт}}^2 K_{\text{т}}} & \frac{1}{2T_{\text{µт}}^2 k_{\text{т}}} \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_a} & 0 & 0 \\ 0 & 0 & \frac{k_{\text{н}}}{T_a K_{\text{д}}} & 0 & 0 & 0 & -\frac{1}{T_a} & 0 \\ 0 & & 0 & 0 & 0 & k_{\text{іс}} & -k_{\text{іс}} & 0 \end{vmatrix}\end{array}$$

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

3.4 Розрахунок перехідних процесів двомасової систем

Сумарний момент інерції двигуна та жорстко з'єднаних з ним елементів приймаємо $J_{\text{д}\Sigma}=1,2J_{\text{д}}=1,2\cdot 7=8,4\ кг\cdot м^2$

Момент інерції вантажу розраховуємо за формулою

$$J_{\text{м}}=\frac{(G_{\text{вп}}+m_0)\cdot V_{\text{п}}^2}{\omega_{\text{н}}^2},$$

$$J_{\text{м}}=\frac{(37000+1000)\cdot 0,1^2}{59,7^2}=0,11\ кг\cdot м^2.$$

Інші параметри системи визначені в попередніх розділах.

Розрахунок перехідних процесів двомасової системи здійснювався із застосуванням пакету прикладних програм MATLAB. Для побудови перехідних характеристик було створено моделі системи в середовищі SIMULINK. На рис. 3.6 представлена схема моделі двомасової системи, побудована відповідно до алгоритмічної схеми рис. 3.4 та рівнянь стану (3.5). Спрощена модель, розроблена за алгоритмічною схемою рис. 3.5 та рівняннями стану (3.6), наведена на рисунку 3.7.

Для моделювання можна також використовувати блок State-Space у SIMULINK (рис. 3.8), в якому задаються імена матриць двомасової системи. Кількість вихідних координат, тобто розмір вектора виходу $\vec{Y}(t)$, визначає кількість вікон Scope для відображення графіків перехідних процесів. При цьому вікно для введення матриць **A**, **B**, **C** і **D** відображається при подвійній активації блоку State-Space (рис. 3.9).

Матриця виходу **C** приймається одиничною, якщо необхідно побудувати графіки для всіх змінних стану. У випадку аналізу частини змінних стану, одиниці розташовуються лише на тих рядках, які відповідають необхідному вектору виходу $\vec{Y}(t)$, тоді як матриця **D** встановлюється нульовою.

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

На рисунках 3.10–3.13 представлені графіки перехідних процесів двомасової системи. Для зручності та зменшення обсягу інформації на екрані та при друку відображалися тільки перші чотири змінні стану. З аналізу графіків видно, що перехідні процеси швидкості механізму (вантажу) та моменту пружності за задавальною дією, а також всі змінні стану за збурюючою дією демонструють характер слабозатухаючих коливань.

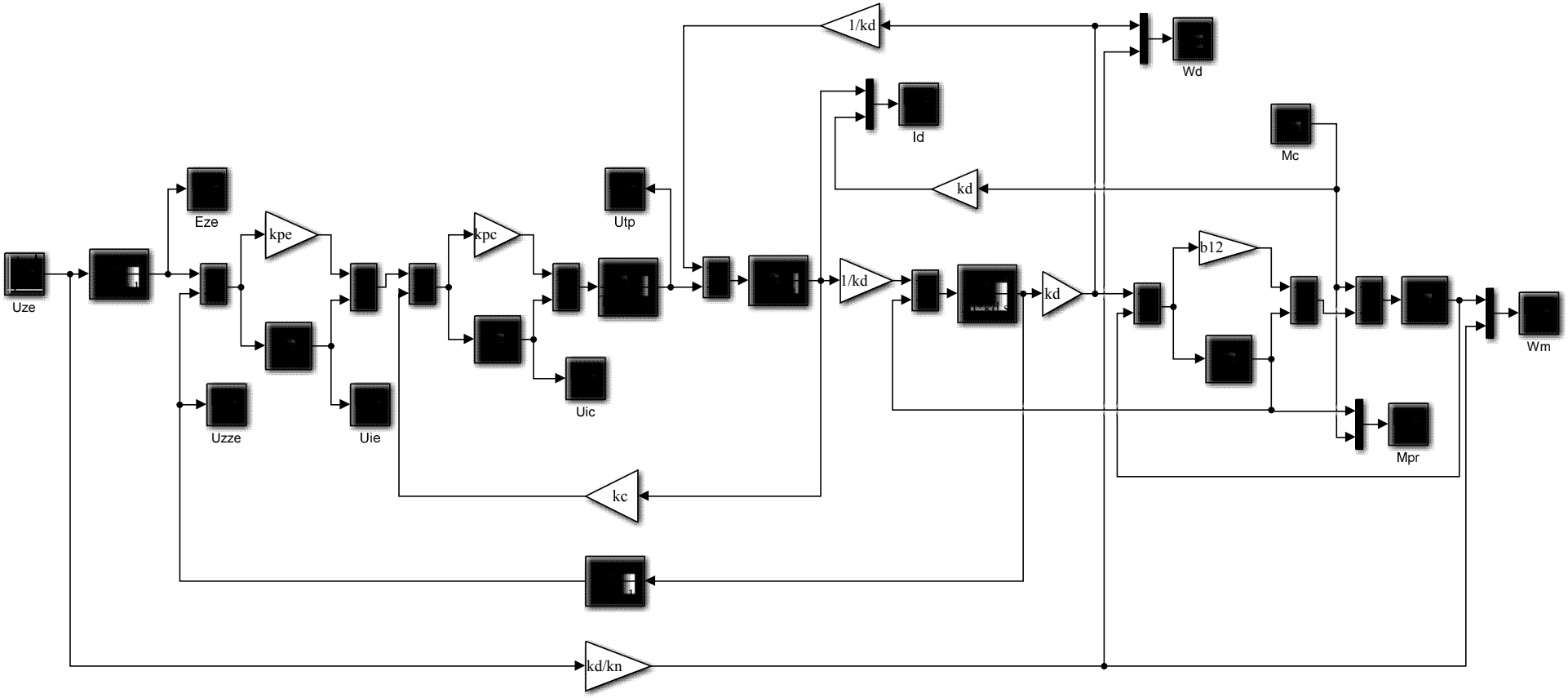


Рисунок 3.6 – Модель двомасової системи в SIMULINK/MATLAB

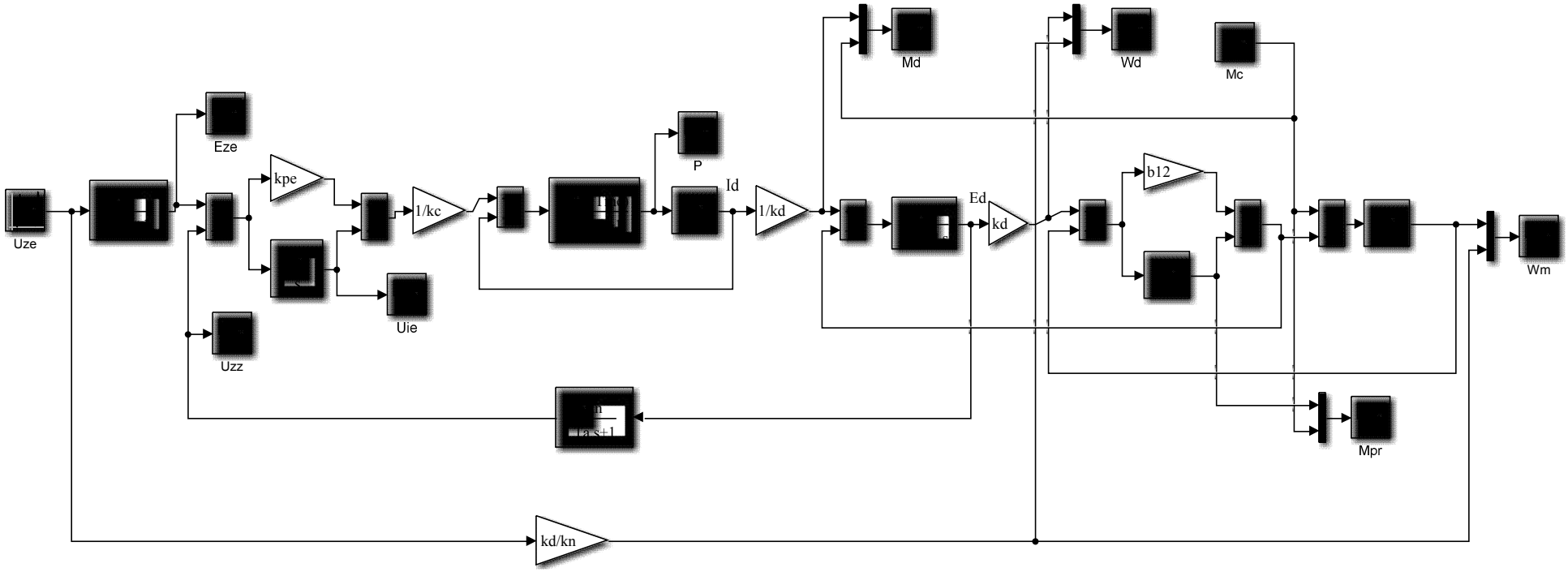


Рисунок 3.7 – Спрощена модель двомасової системи в SIMULINK/MATLAB

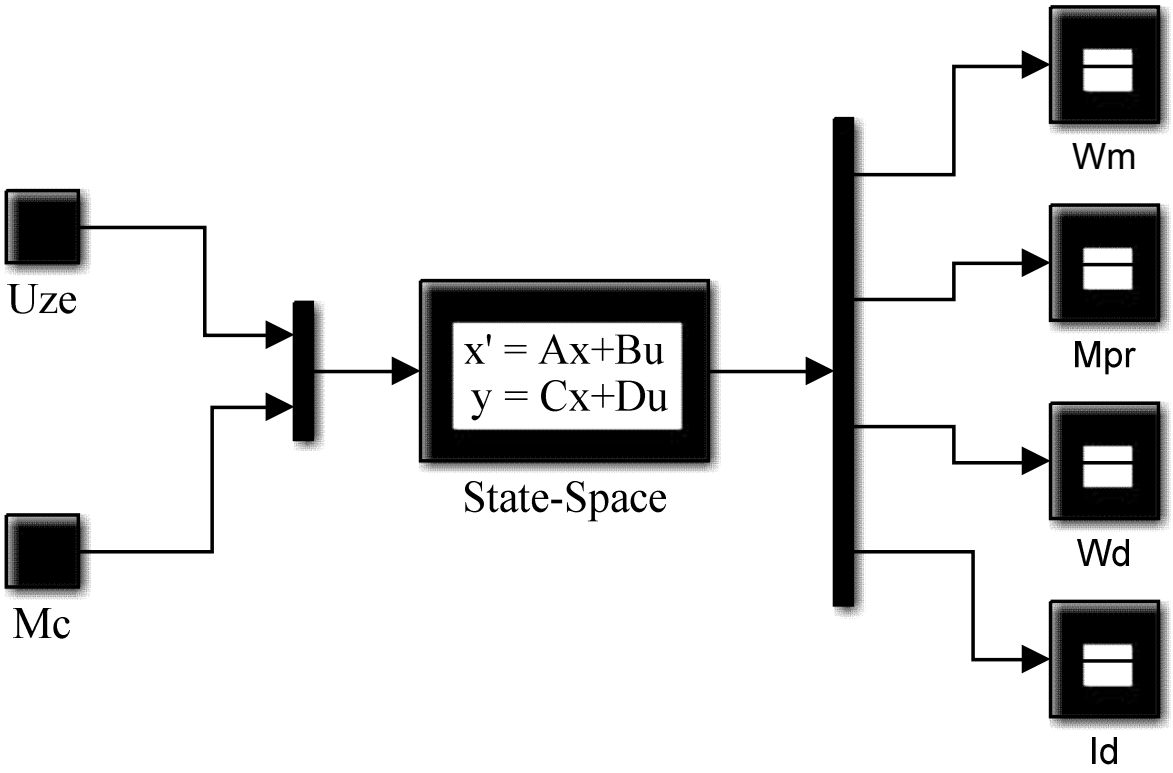


Рисунок 3.8 – Модель системи з використанням блоку State-Space

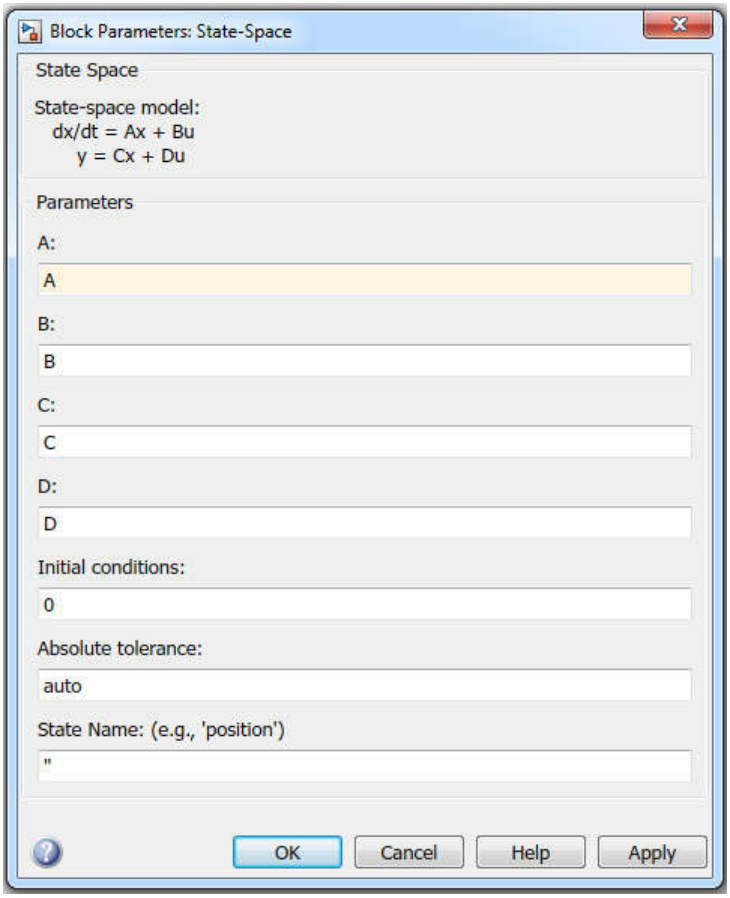
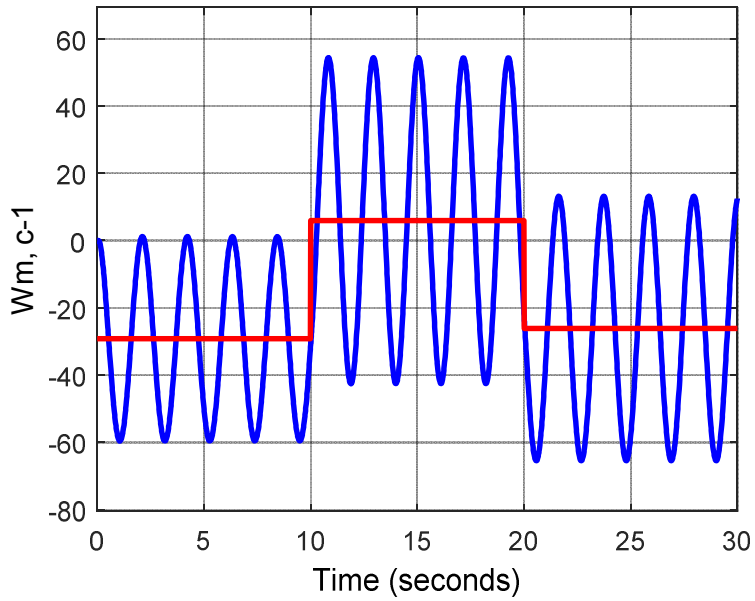
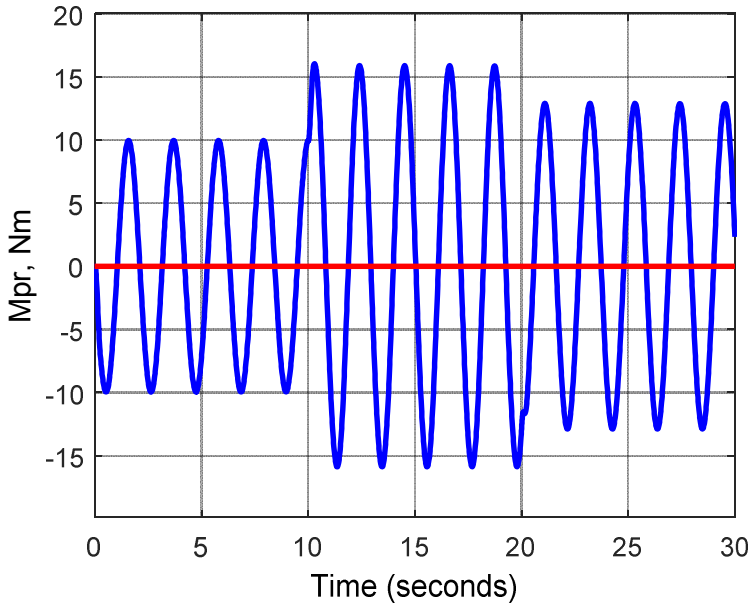


Рисунок 3.9 – Вікно завдання матриць **A**, **B**, **C**, **D** блоку State-Spase

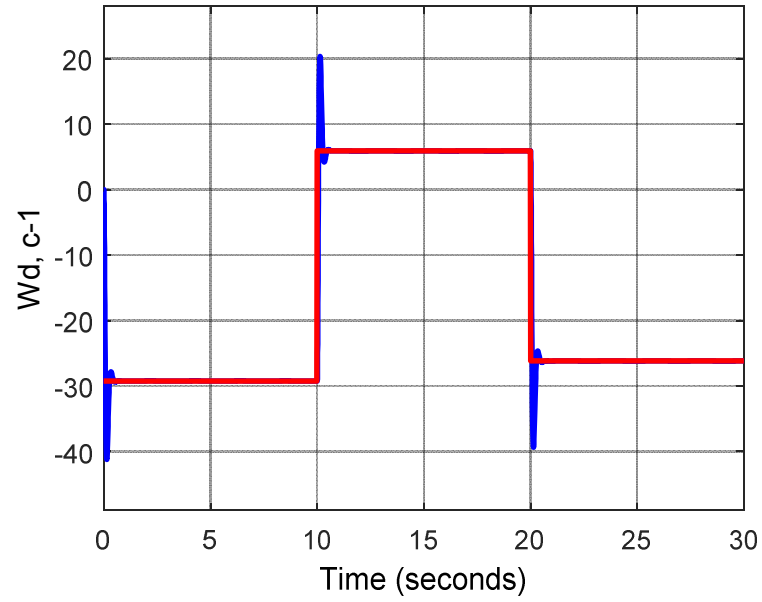


а)

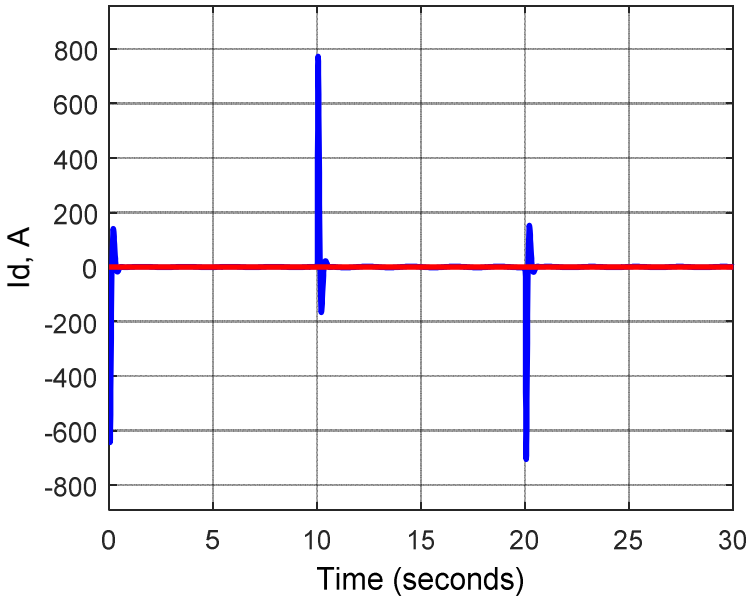


б)

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

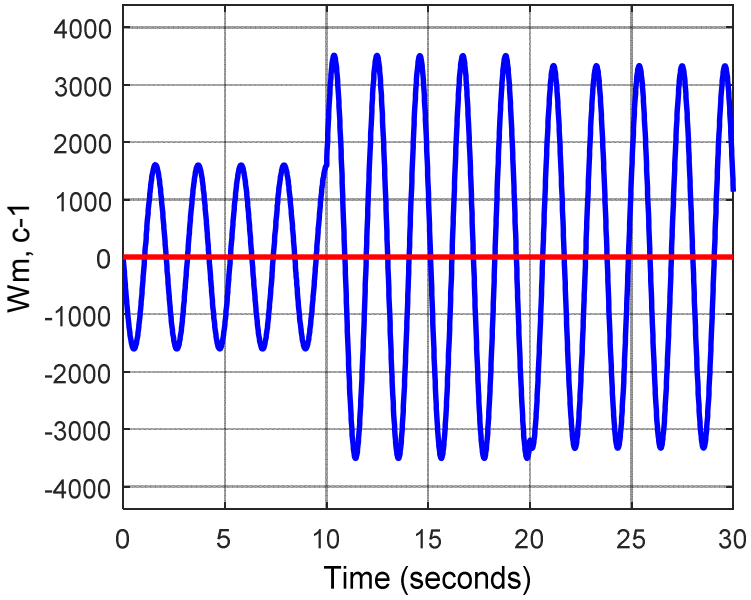


а)

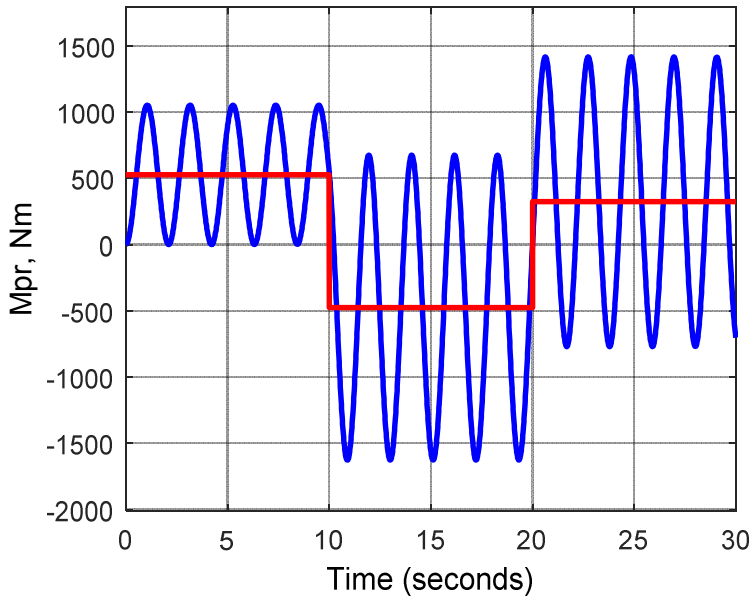


б)

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

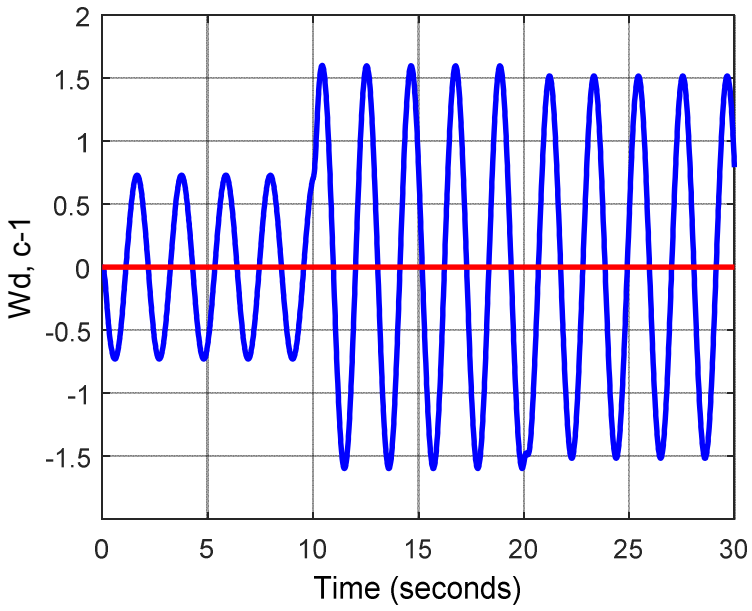


а)

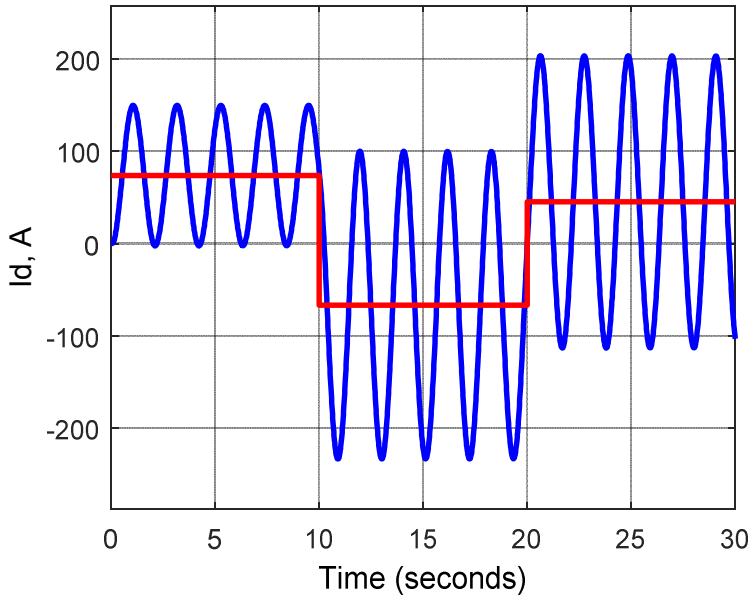


б)

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



а)



б)

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

ВИСНОВКИ ДО РОЗДІЛУ 3

1. Виконано дослідження роботи підйомного механізму мостового крана з урахуванням кінцевої жорсткості підйомного канату. Проаналізовано кінематичну схему електроприводу механізму підйому. Встановлено, що механічна частина системи є набором зв'язаних мас, які рухаються з різними поступальними та обертальними швидкостями. Під навантаженням елементи системи зазнають деформацій через пружність зв'язків. Зміни навантаження спричиняють відносні переміщення мас, величина яких визначається жорсткістю механічних зв'язків. Аналіз показав, що розрахункові механічні схеми, після виділення основних мас та ключових жорсткостей, спрощуються до трьохмасових або двомасових моделей. Двомасова пружна механічна система є найбільш зручною для дослідження впливу пружних зв'язків на динаміку електроприводу і тому обрана як основний об'єкт моделювання.

2. Розроблено математичну модель двомасової системи. Отримано рівняння стану системи та її представлення у векторно-матричній формі. Визначено вектор стану, що включає змінні стану системи, вектор зовнішніх впливів, а також матриці стану AAA та керування BBB. Створено алгоритмічну схему двомасової системи, а також її спрощену версію з оптимізованим по модульному критерію контуром струму.

3. Проведено моделювання двомасової системи на ЕОМ за допомогою пакету MATLAB. У середовищі SIMULINK побудовано модель системи, яка дозволяє досліджувати динаміку змінних стану. В якості тестового сигналу на вхід подавали ступінчастий сигнал з випадковою амплітудою в межах ±24 В, що відповідає зміні швидкості в межах номінальних значень. Результати моделювання показали, що перехідні процеси змінних стану характеризуються коливальним, недостатньо затухаючим характером.

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).

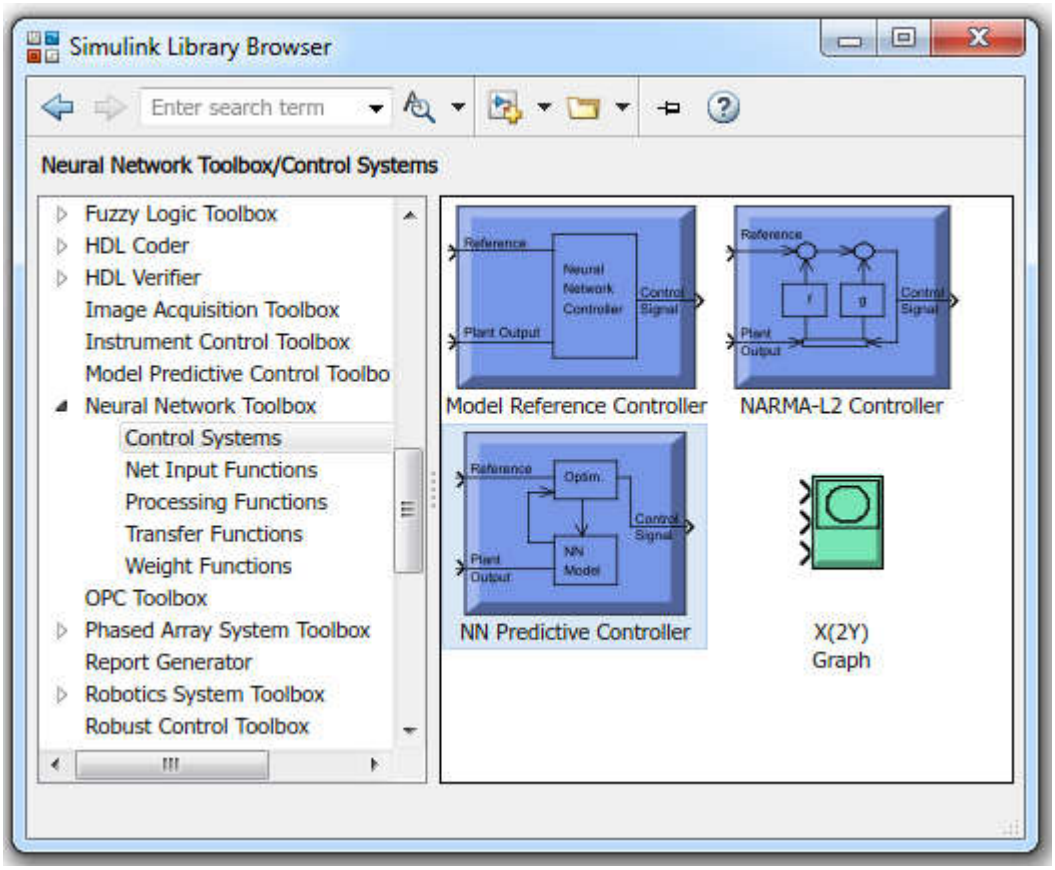


Рисунок 4.1 – Набір блоків Neural Network Blocksets підсистеми Control Systems середовища SIMULINK

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 керува

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.

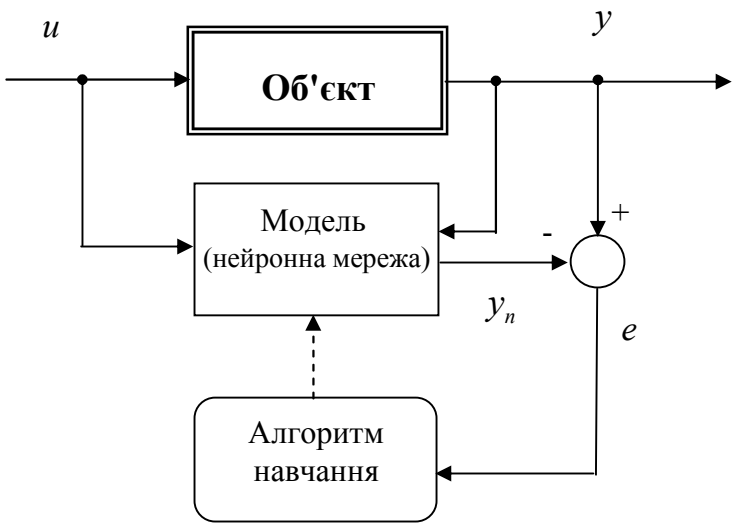


Рисунок 4.2 – Структурна схема підсистеми ідентифікації

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**.

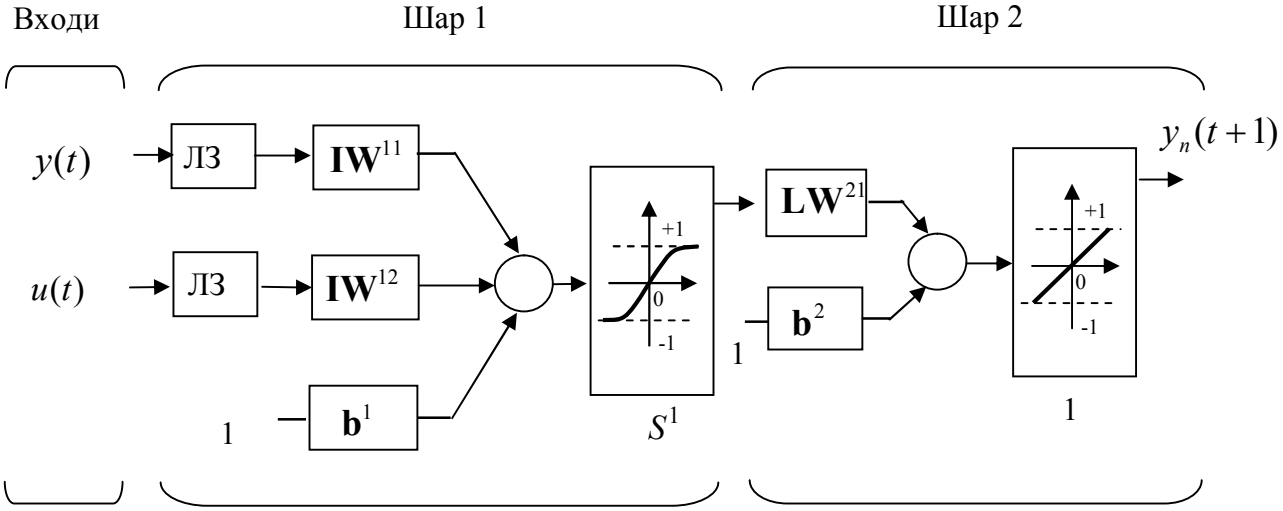


Рисунок 4.3 – Структура нейронної мережі, що моделює об’єкт керування

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.

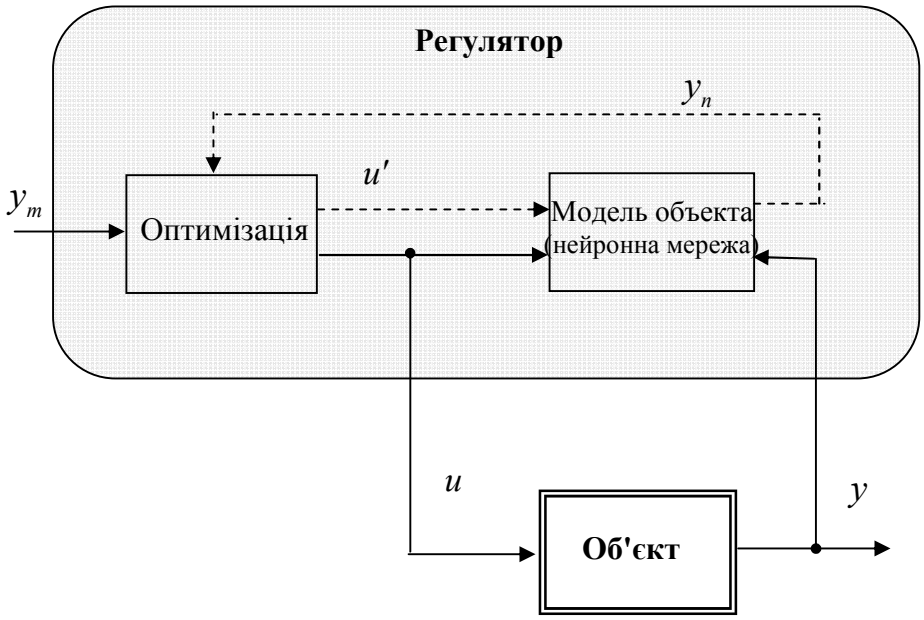


Рисунок 4.4 – Узагальнена схема керування з використанням прогнозуючого регулятора

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.

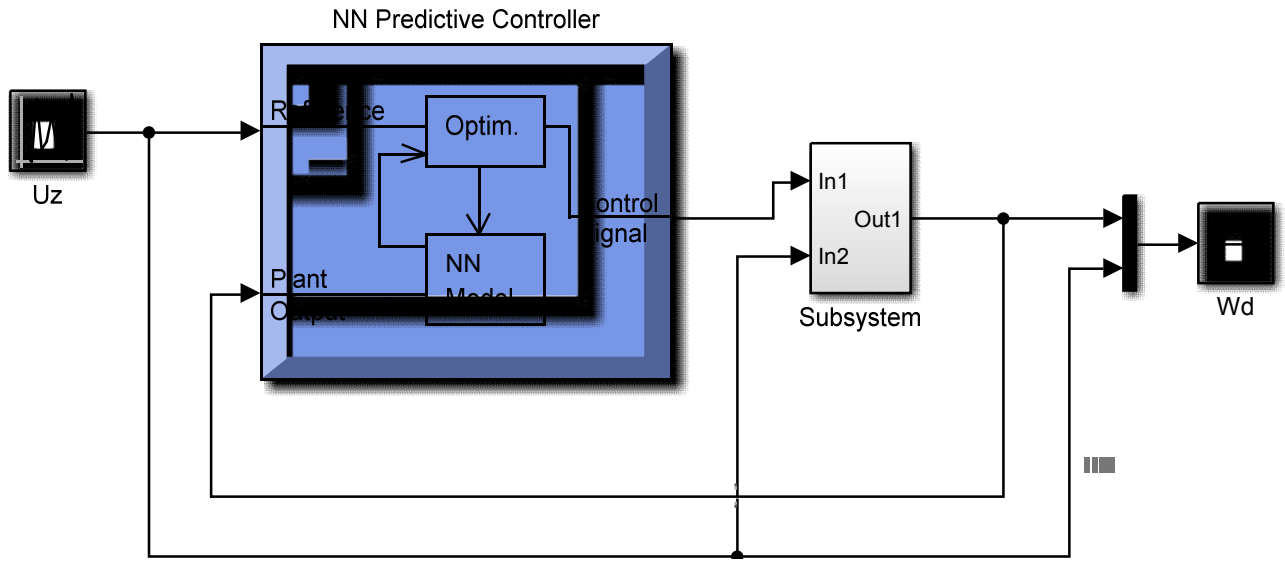


Рисунок 4.5 – Структурна модель системи керування з нейрорегулятором NN Predictive Controller

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.

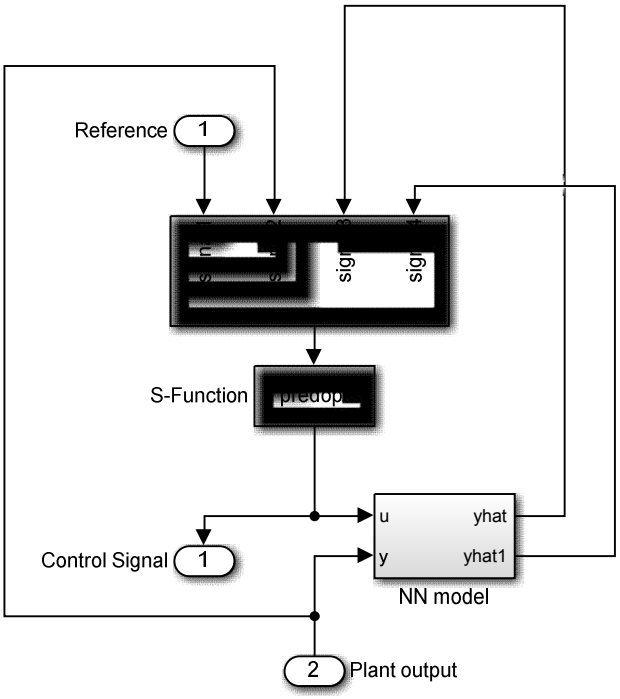


Рисунок 4.6 – Структурна схема нейрорегулятора

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**.

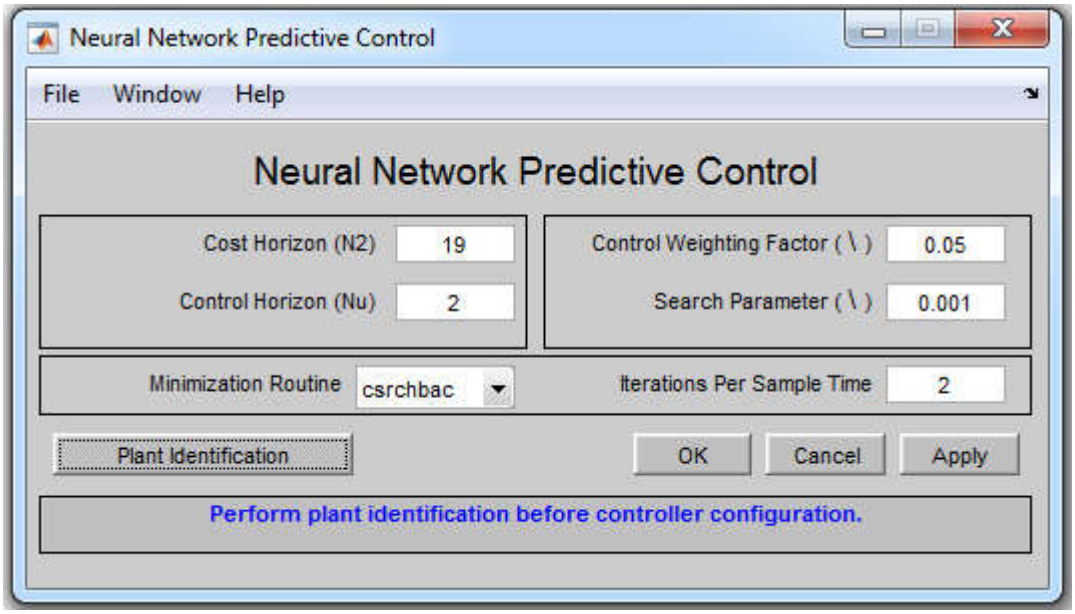


Рисунок 4.7 – Інтерфейс налаштування параметрів регулятора

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.

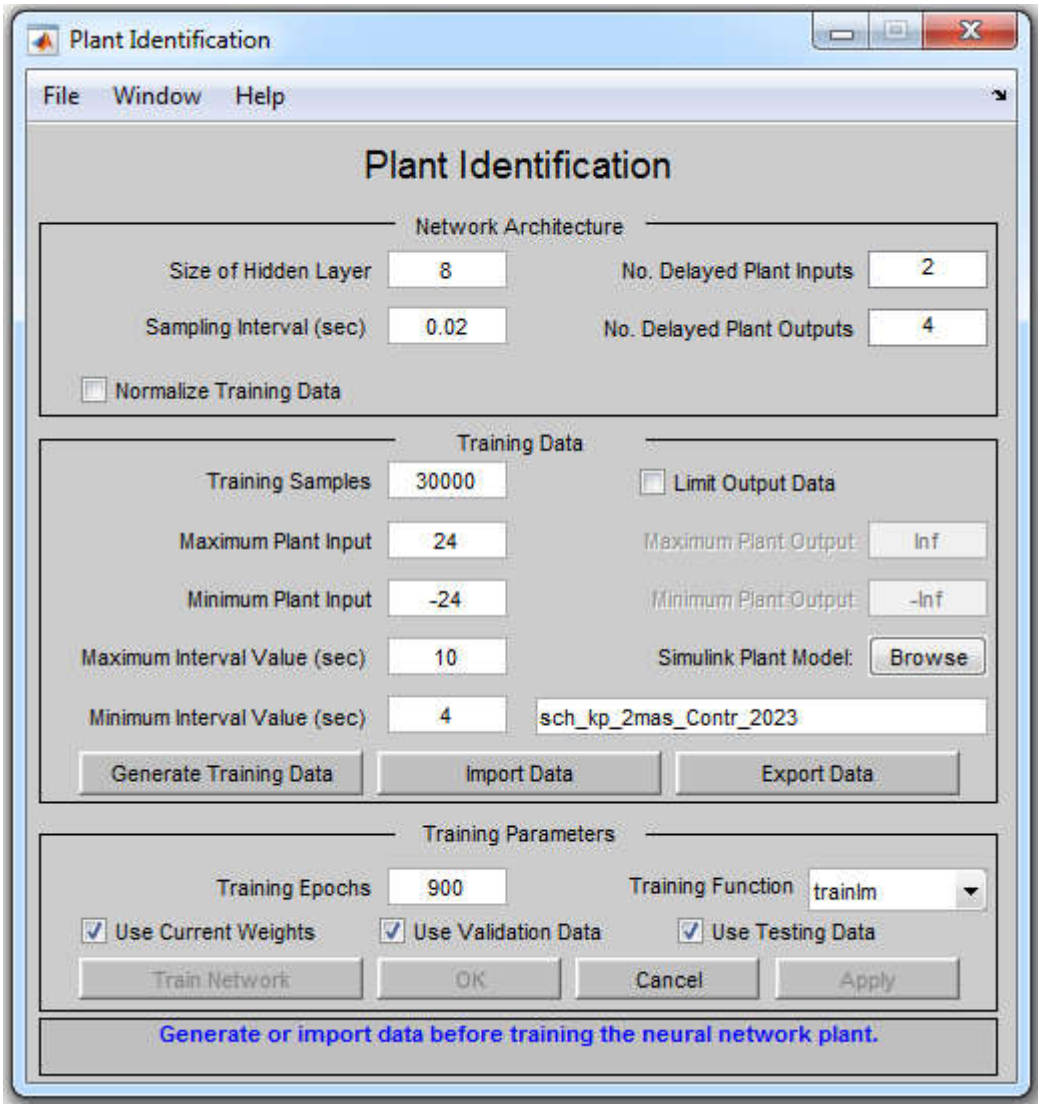


Рисунок 4.8 – Вікно процедури ідентифікації об’єкта керування

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.

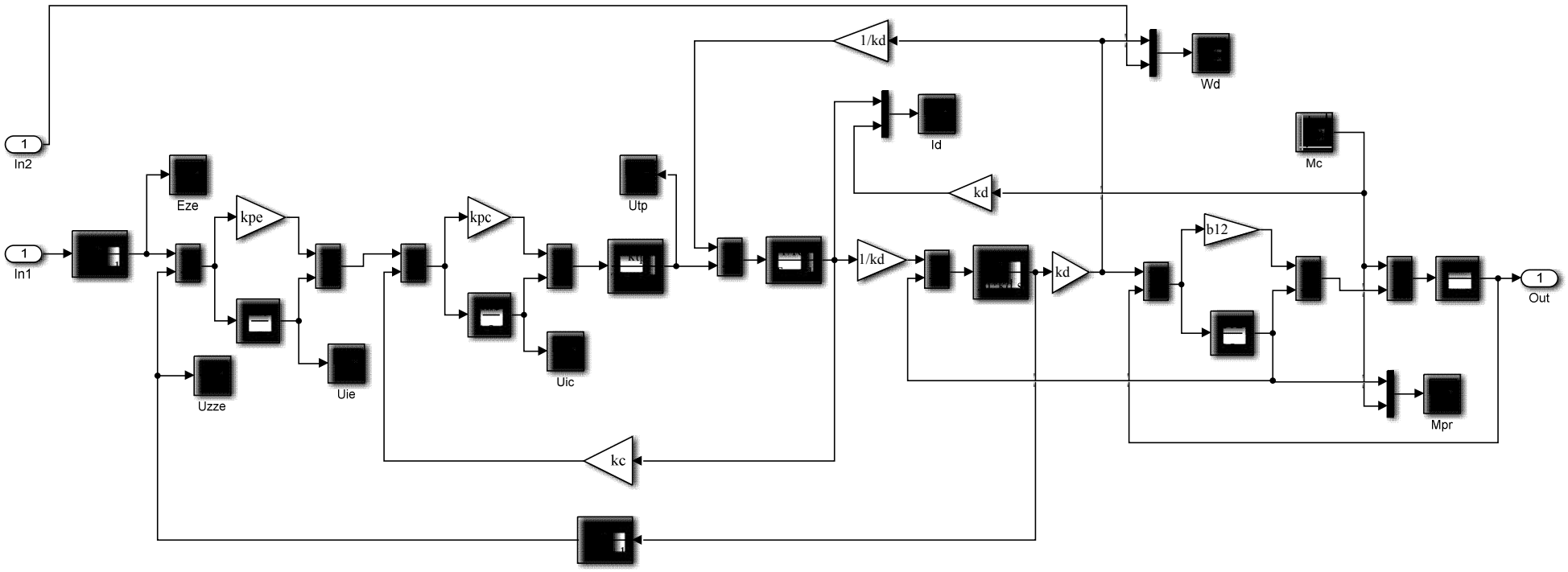


Рисунок 4.9 – Модель керованого об’єкта нейрорегулятора

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.

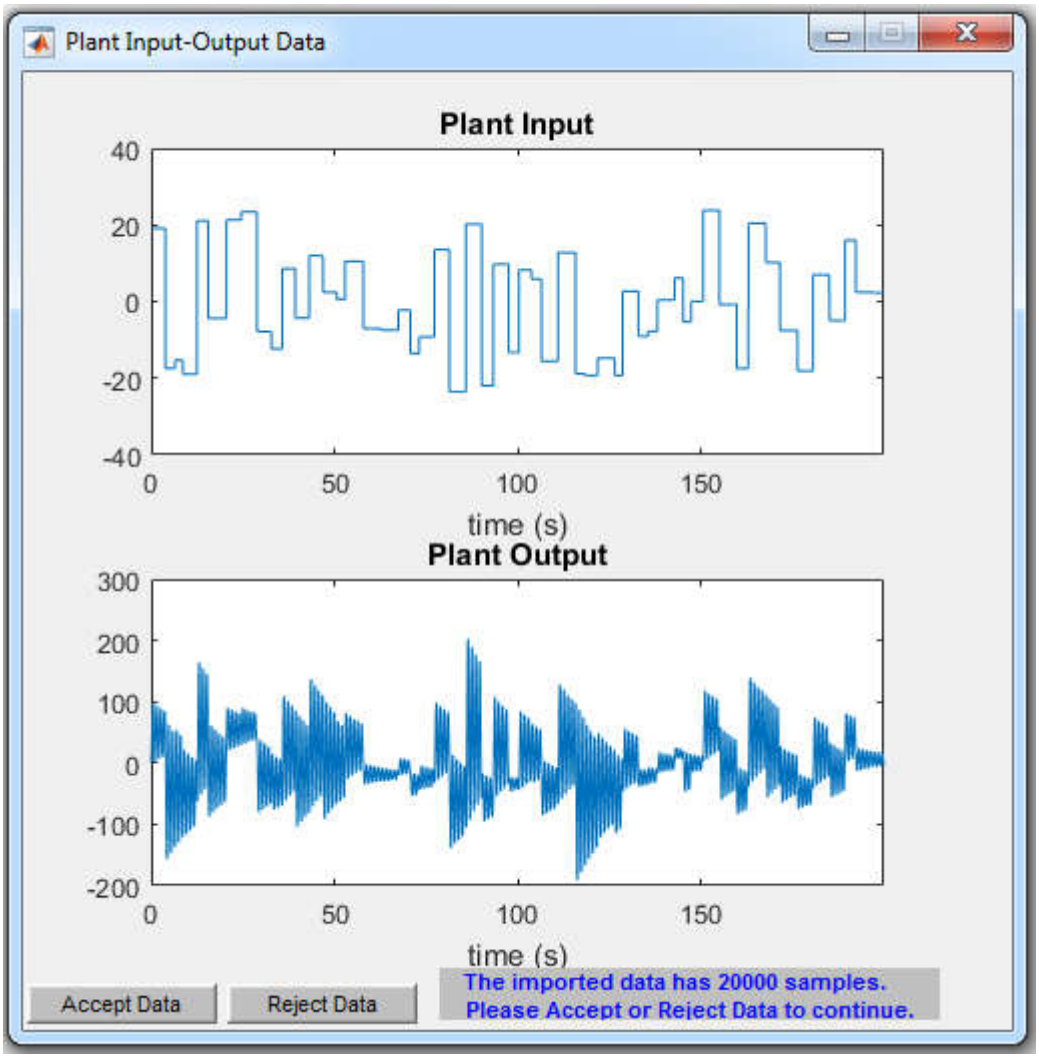


Рисунок 4.10 – Графіки вхідного та вихідного сигналів під час генерації навчальної послідовності

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.

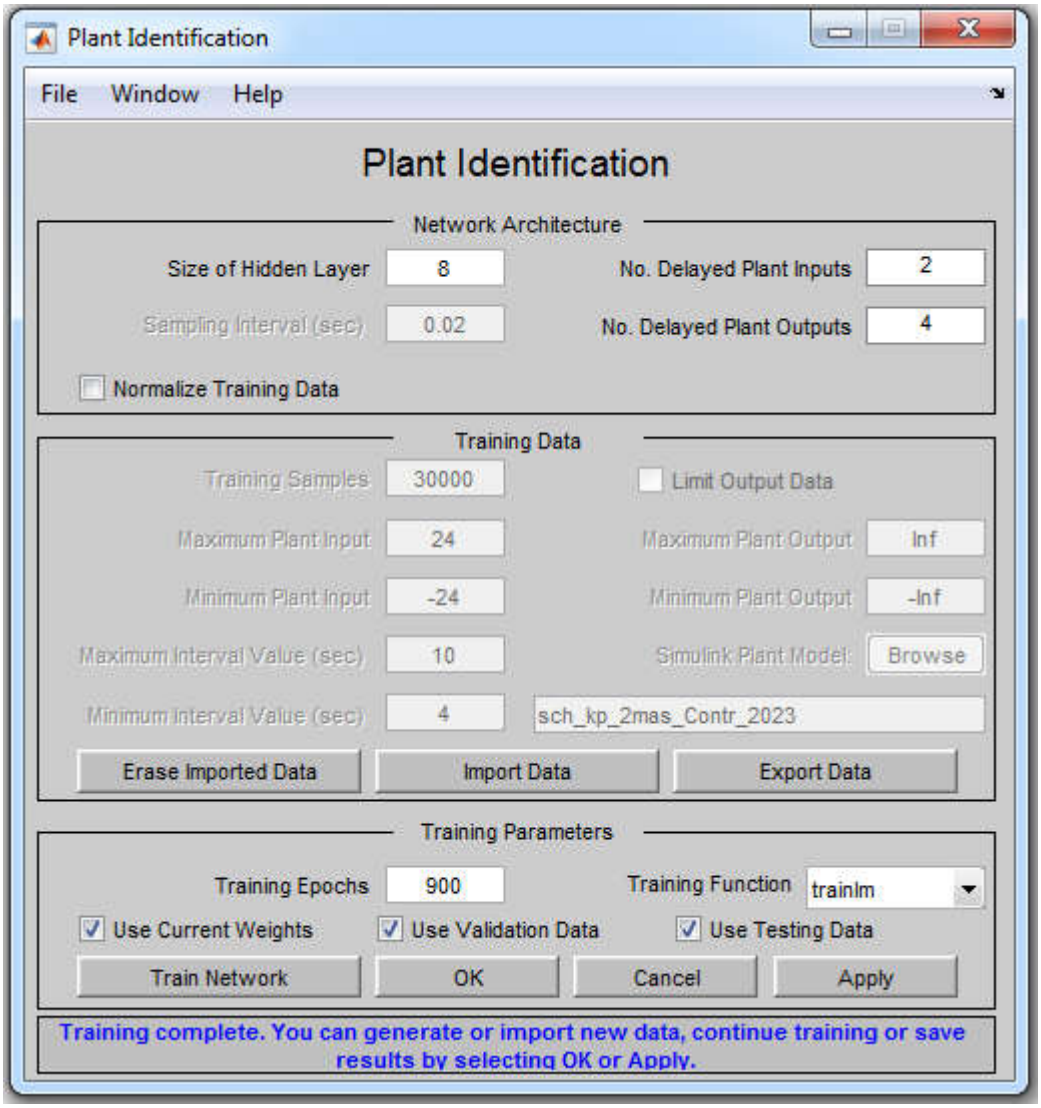


Рисунок 4.11 – Вікно Plant Identification після генерації навчальної послідовності

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).

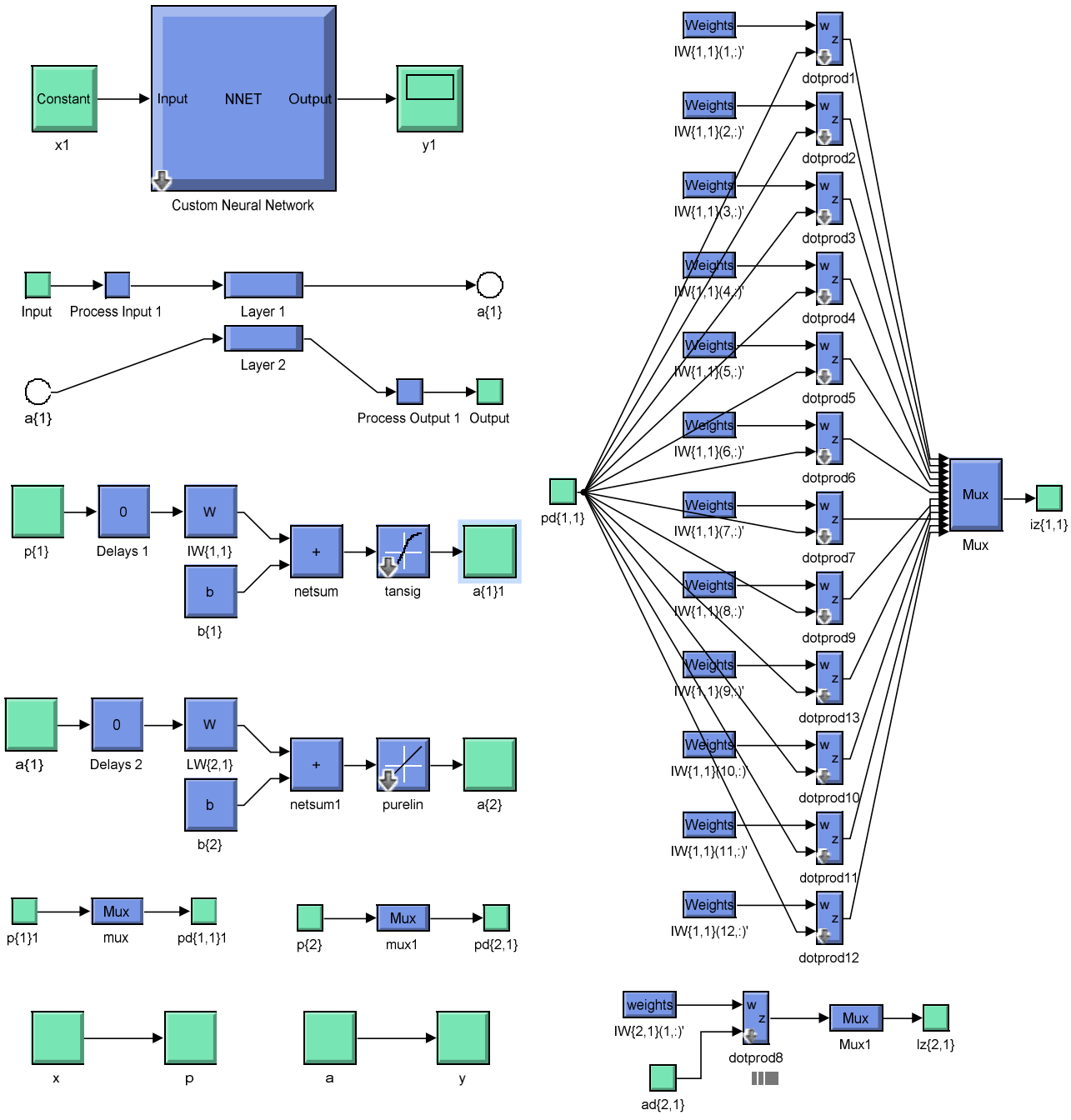


Рисунок 4.12 – Моделі елементів мережі з прямою передачею сигналу, реалізовані в Simulink

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.

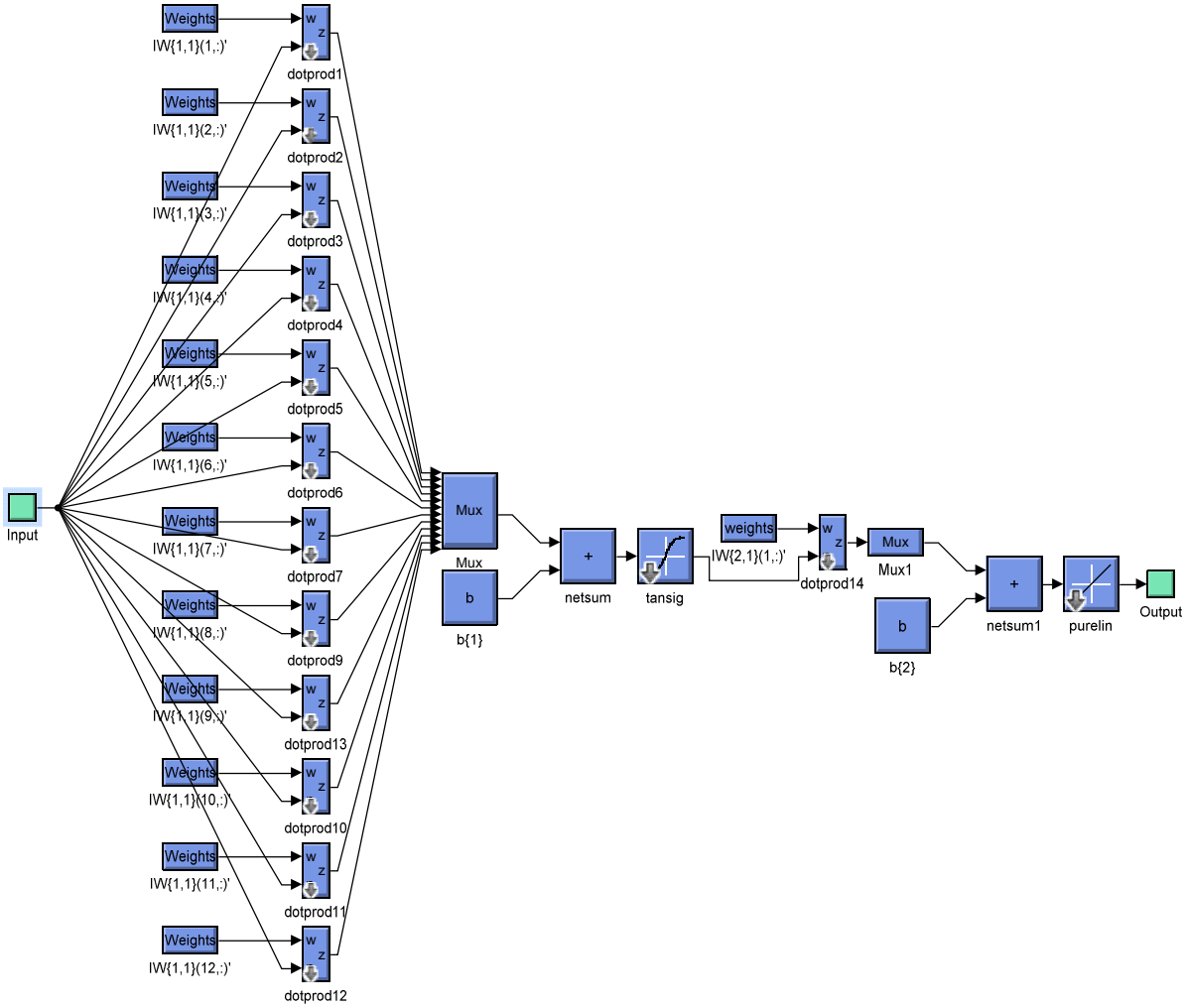


Рисунок 4.13 – Модель статичної мережі з прямою передачею сигналу, побудована в Simulink

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.

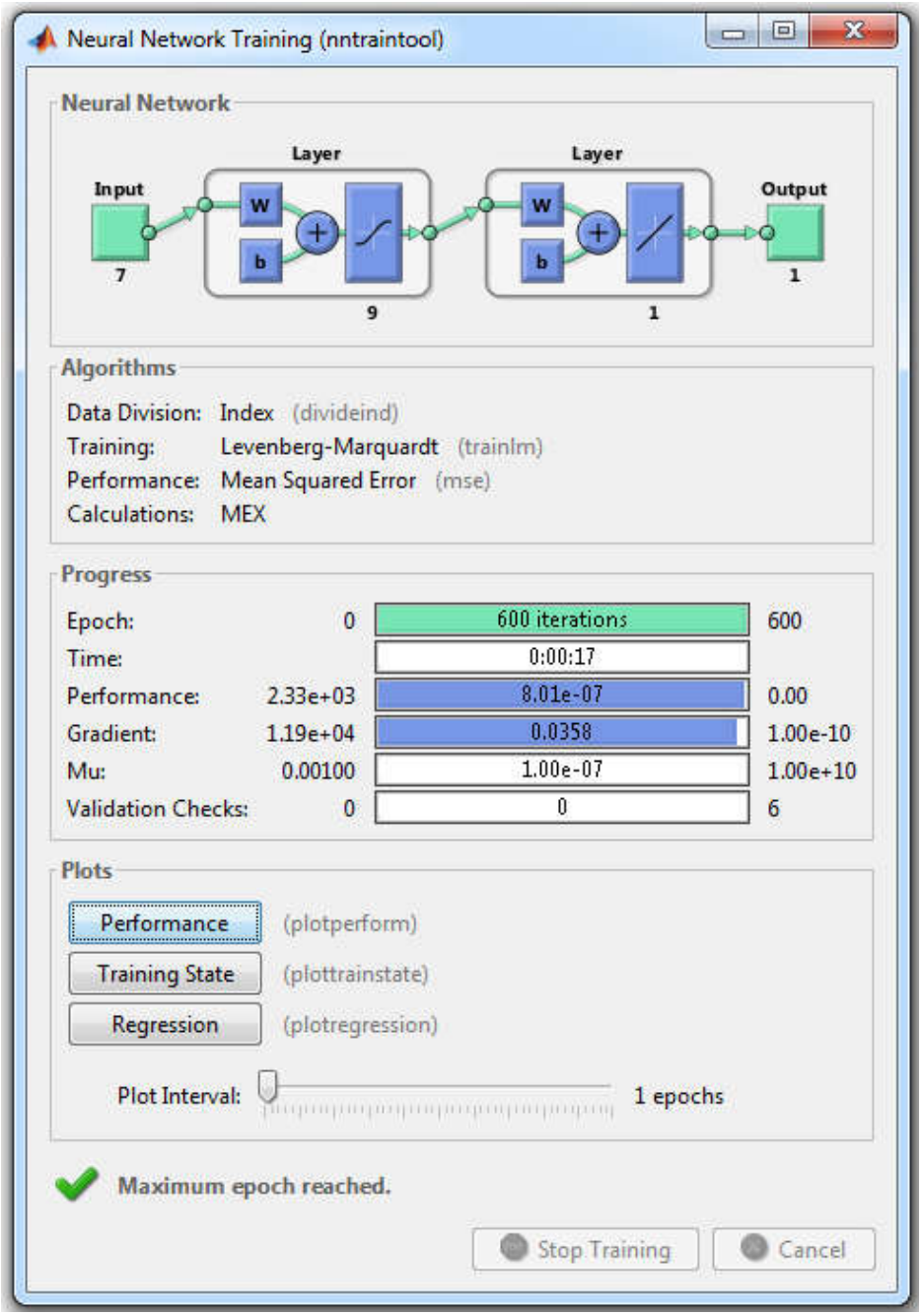


Рисунок 4.14 - Головне вікно для відображення процесу та результатів навчання нейронної мережі

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.

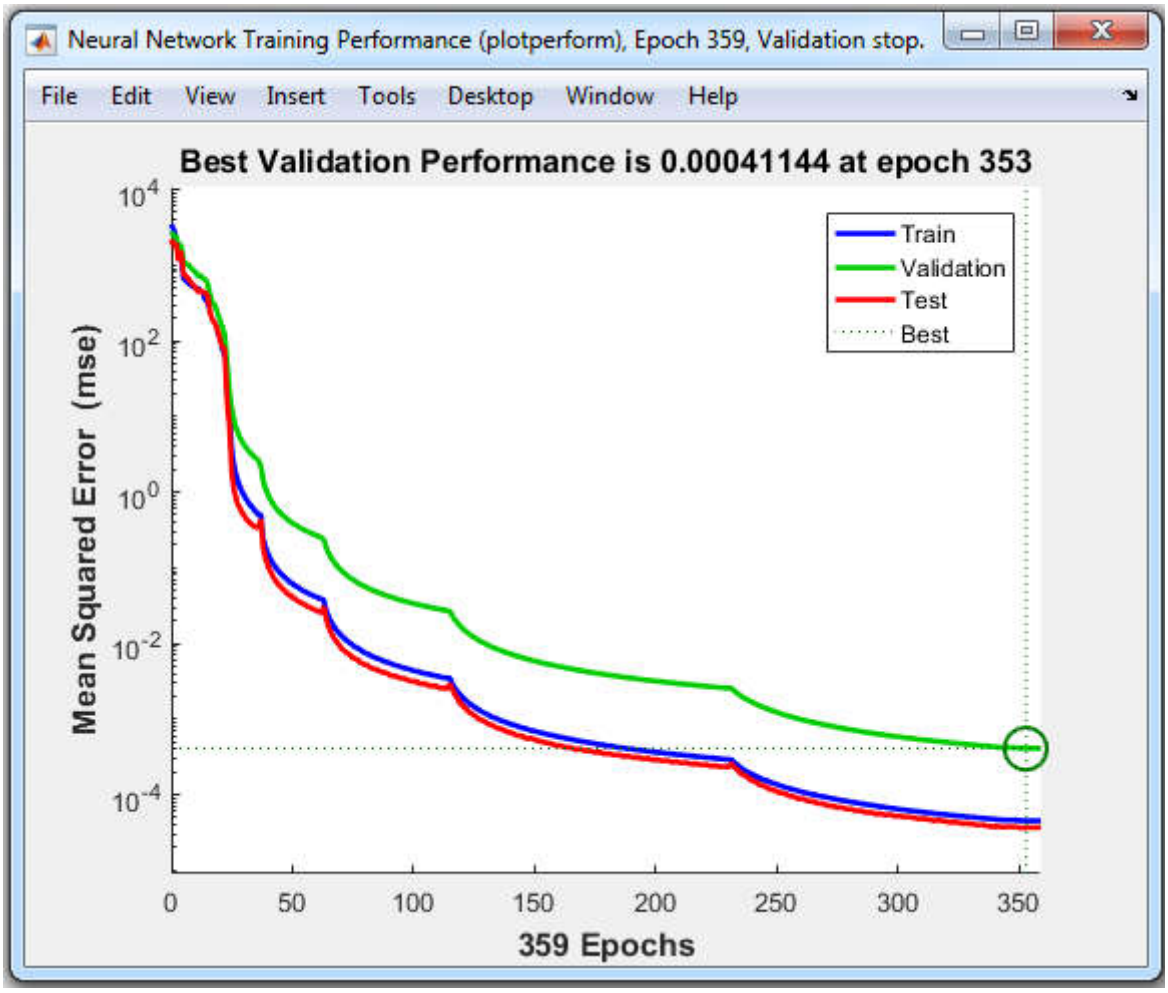


Рисунок 4.15 – Вікно контролю процесу навчання

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.

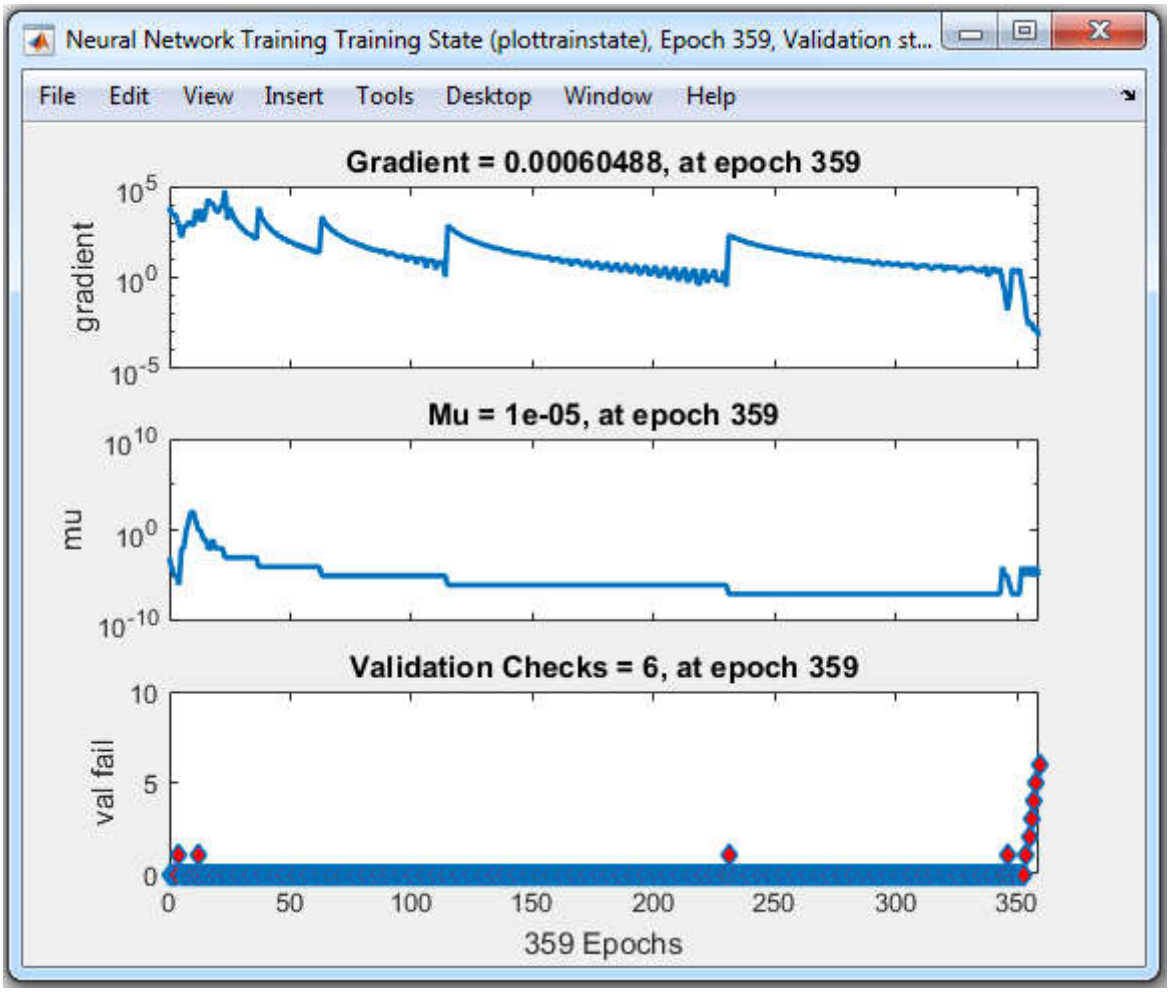


Рисунок 4.16 - Вікно Neural Network Training State

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.

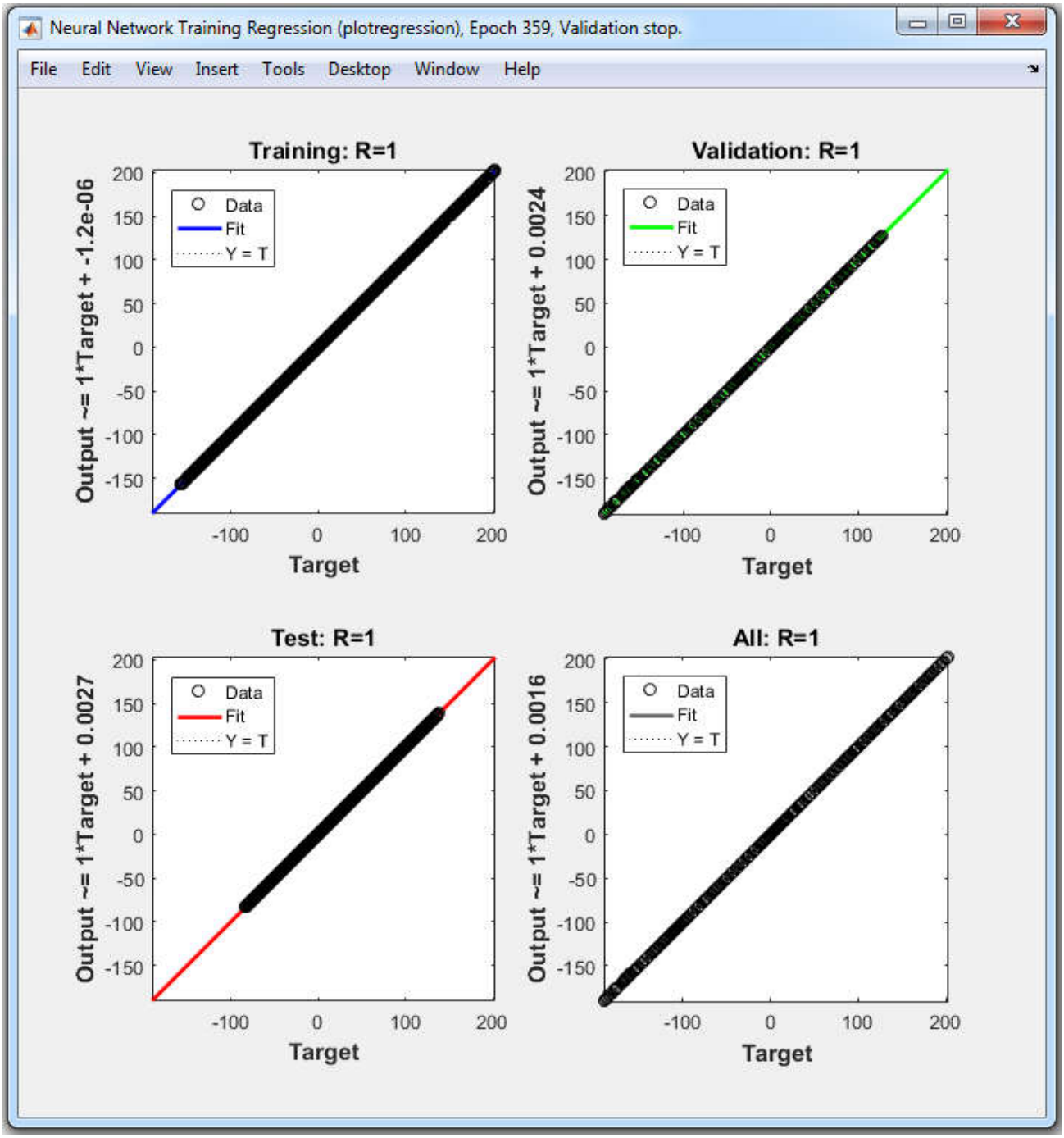


Рисунок 4.17 - Вікно Neural Network Training Regression

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).

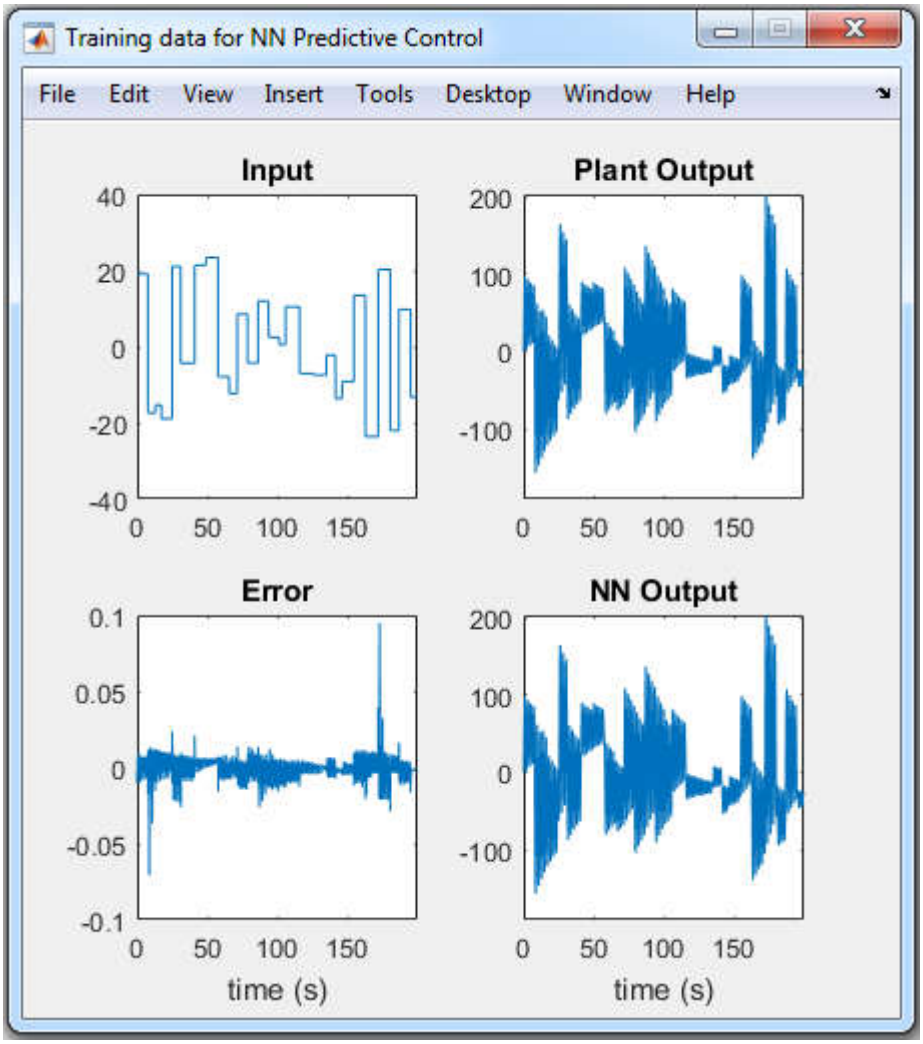


Рисунок 4.18 – Графічне представлення результатів навчання нейронної мережі

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.

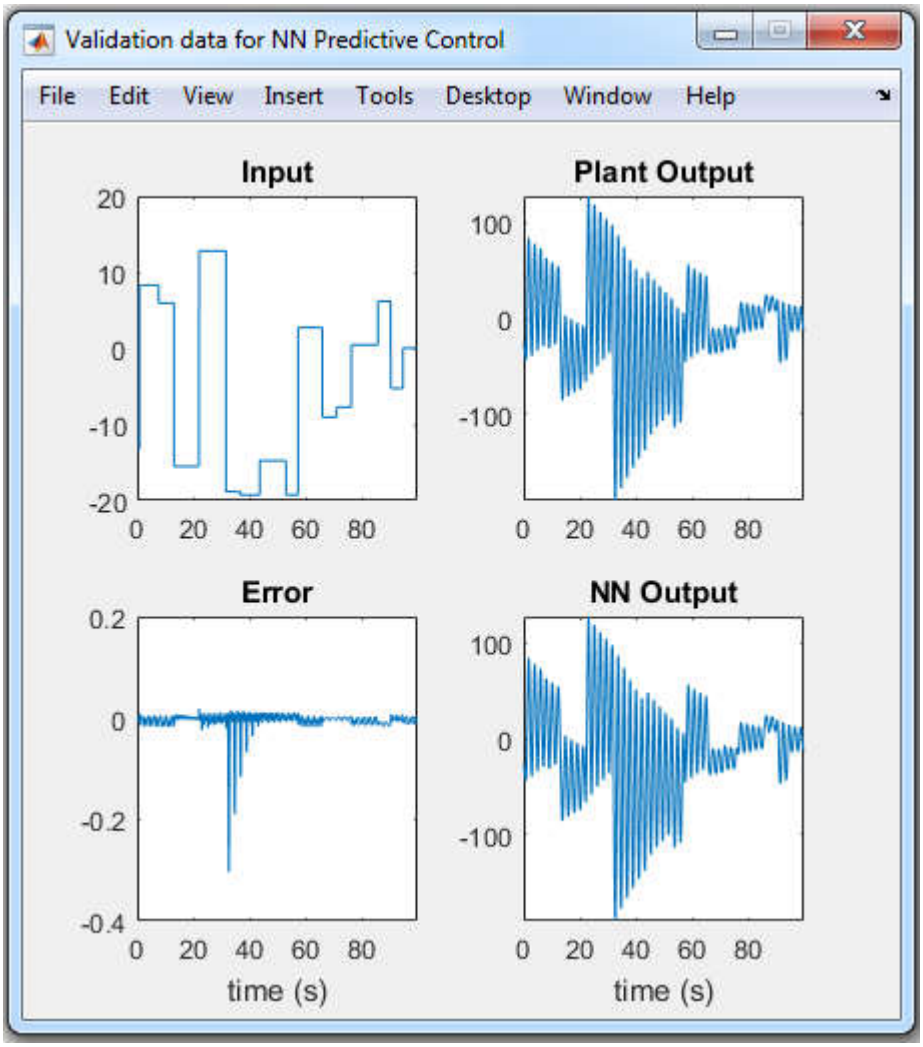


Рисунок 4.19 – Візуалізація результатів перевірки нейронної мережі на контрольній множині

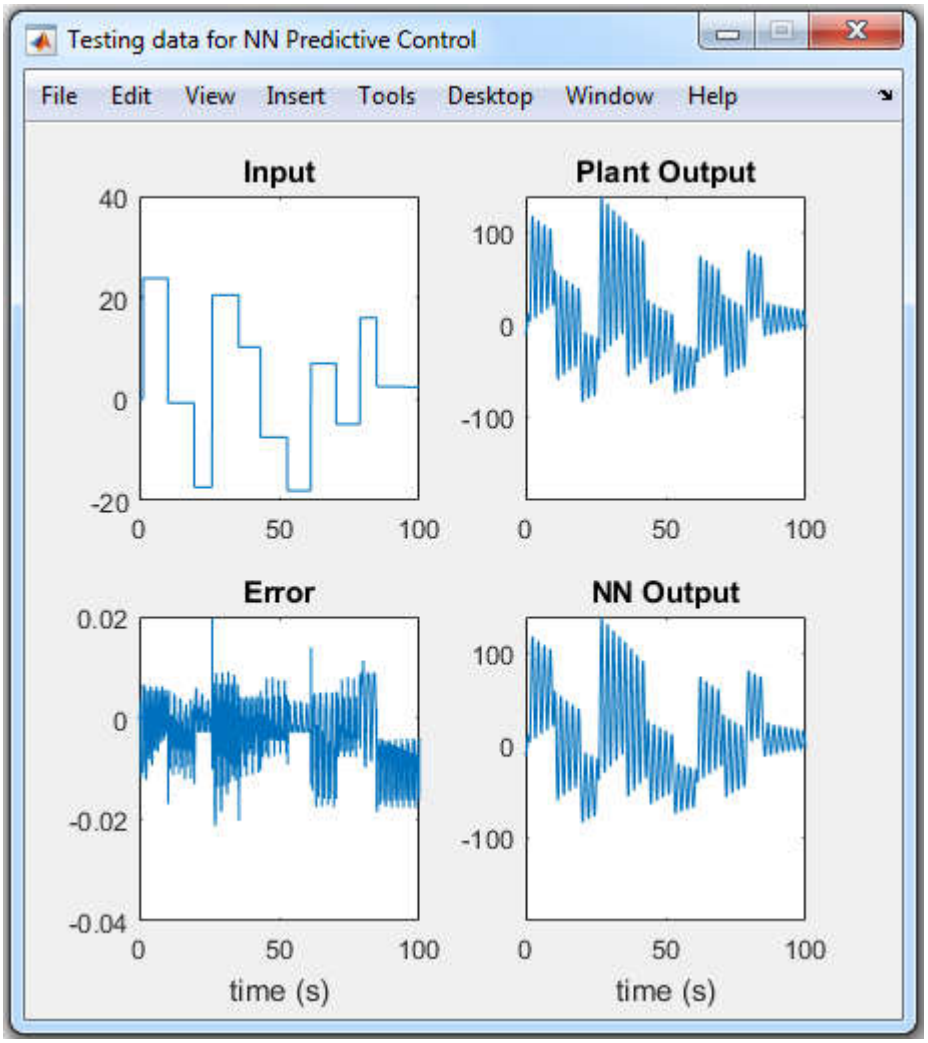


Рисунок 4.20 – Візуалізація результатів перевірки нейронної мережі на тестовій множині

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.

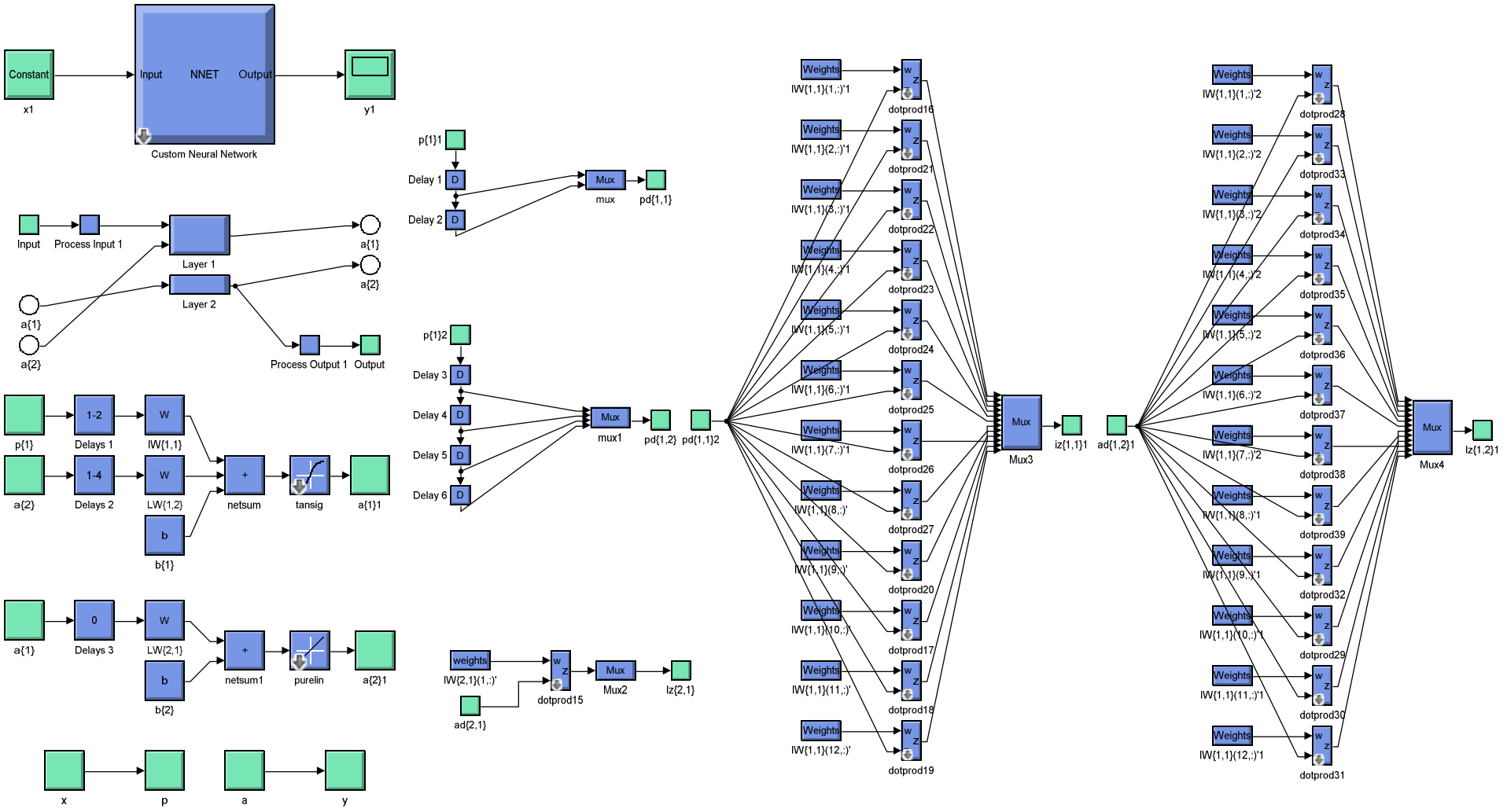


Рисунок 4.21 – Моделі елементів динамічної мережі з блоками затримок у Simulink

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.

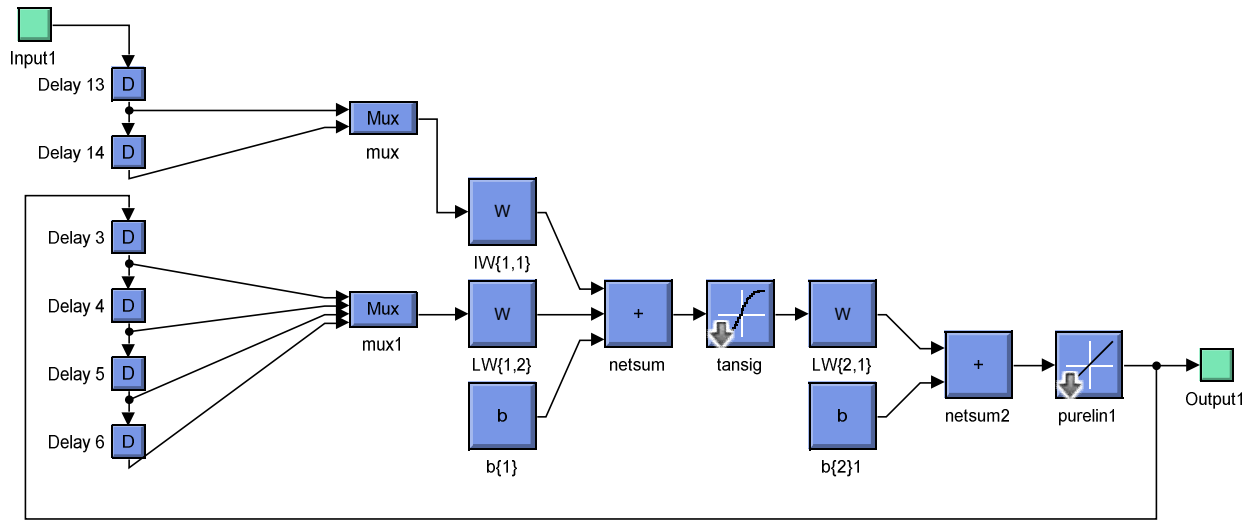


Рисунок 4.22 – Модель динамічної мережі з елементами затримок, побудованої в Simulink

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.

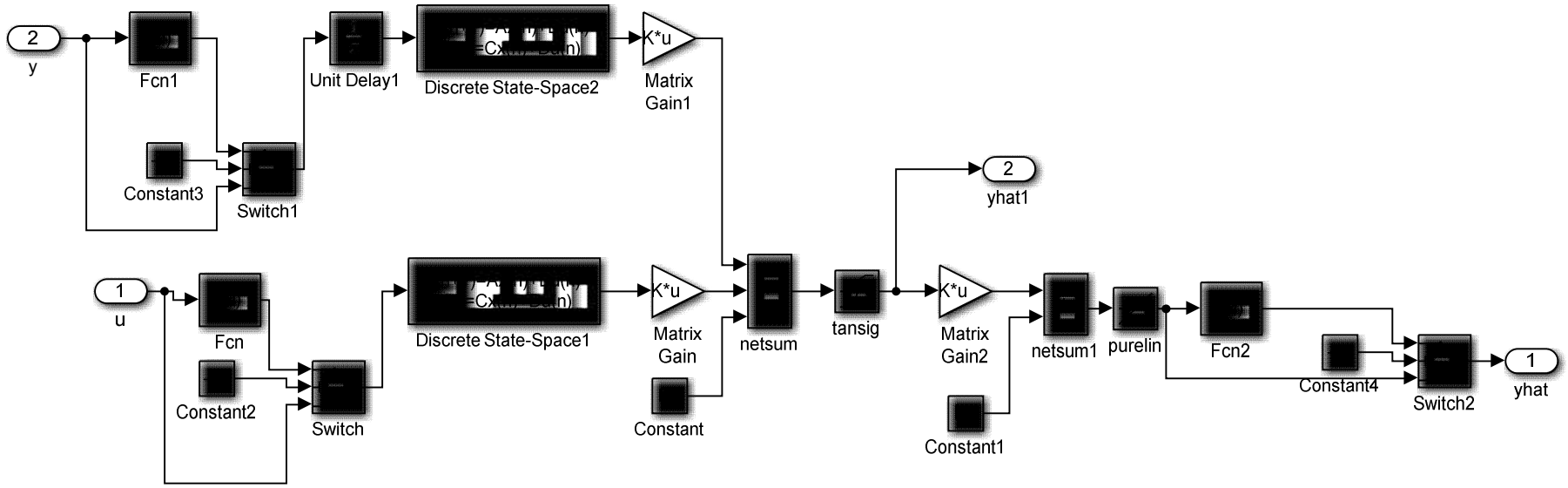


Рисунок 4.23 – Структурна схема нейромережевої моделі об'єкту регулювання

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.

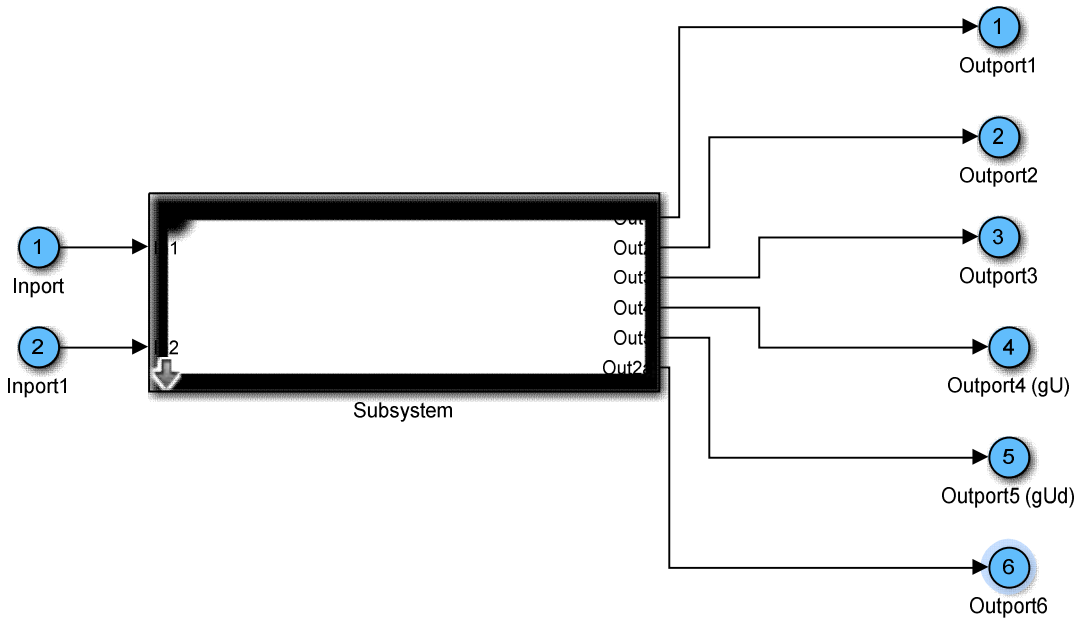


Рисунок 4.24 – Схема ptest3sim2, що використовується для прогнозування процесу функцією predopt

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.

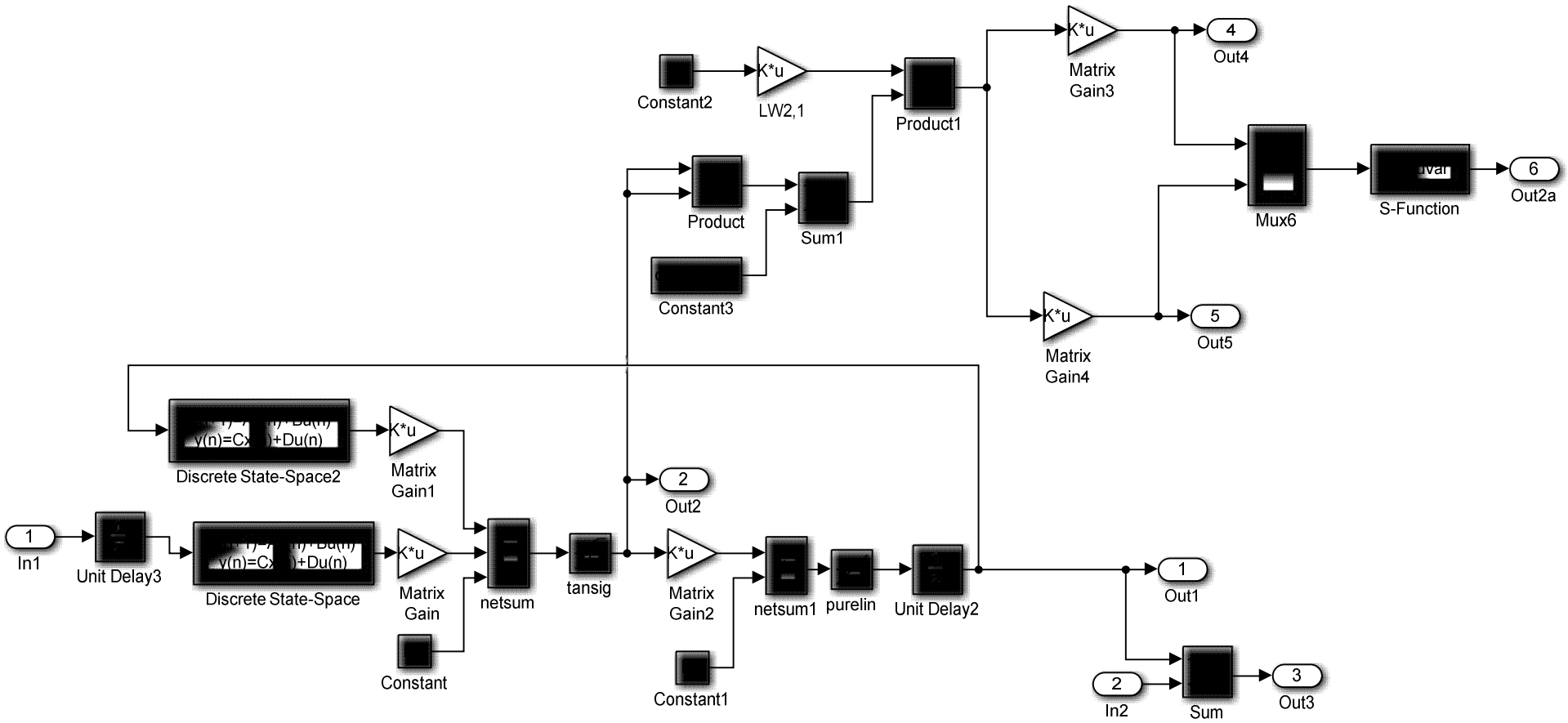


Рисунок 4.25 – Схема блоку Subsystem системи ptest3sim2

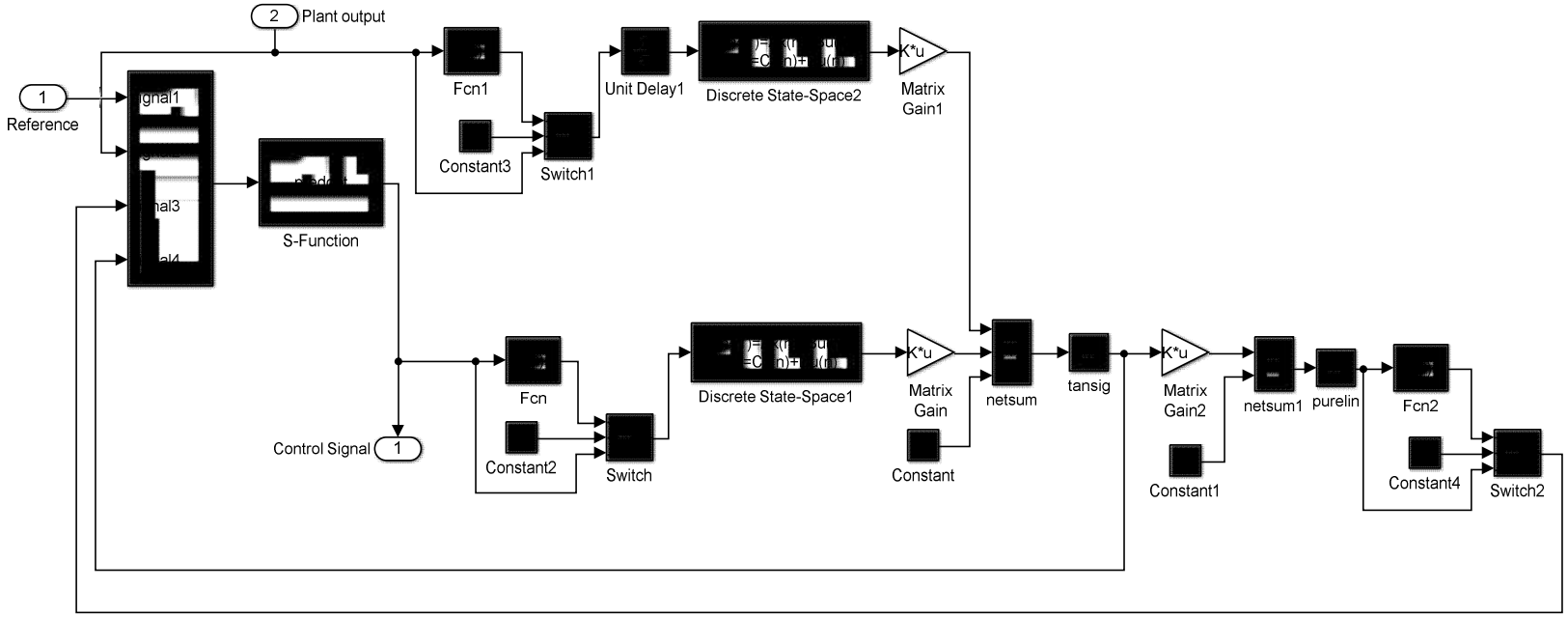


Рисунок 4.26 – Перетворена структурна схема нейрорегулятора

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: $N_{\text{в}} = 10000$, $\Delta t = 0,05 \div 0,1\ c$. 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 $300 \div 600$.

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.

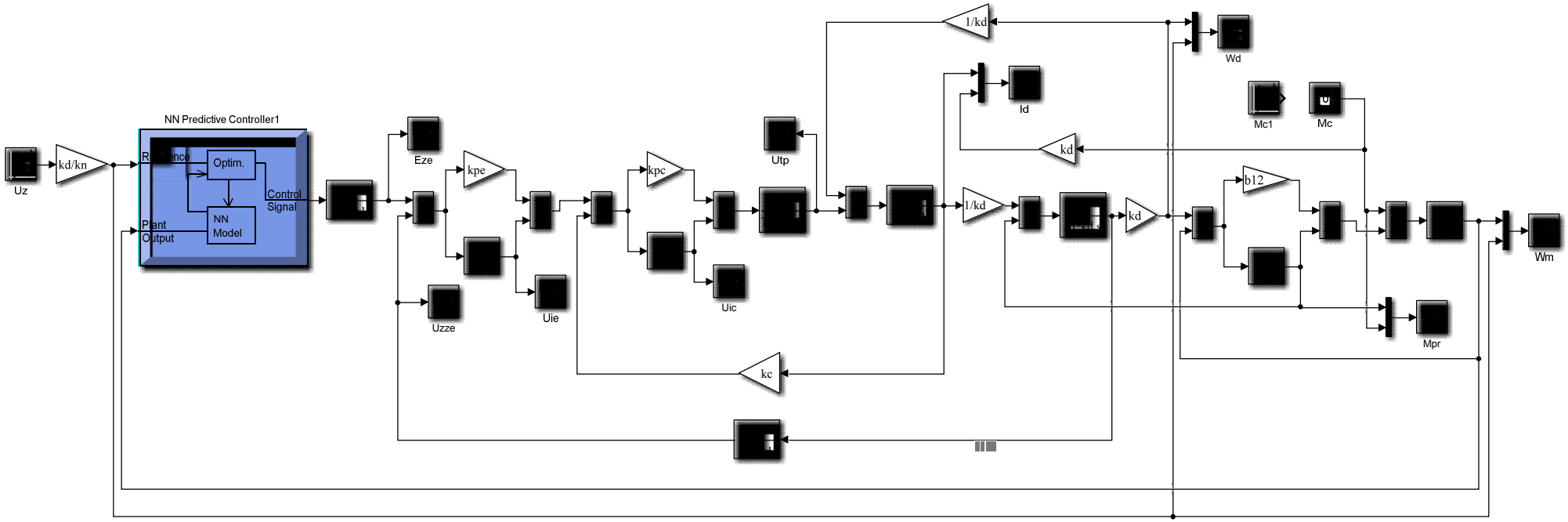
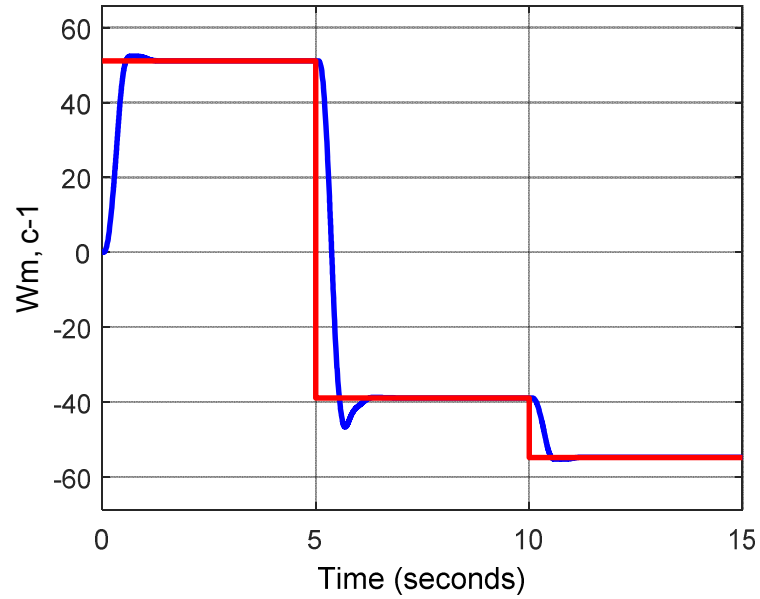
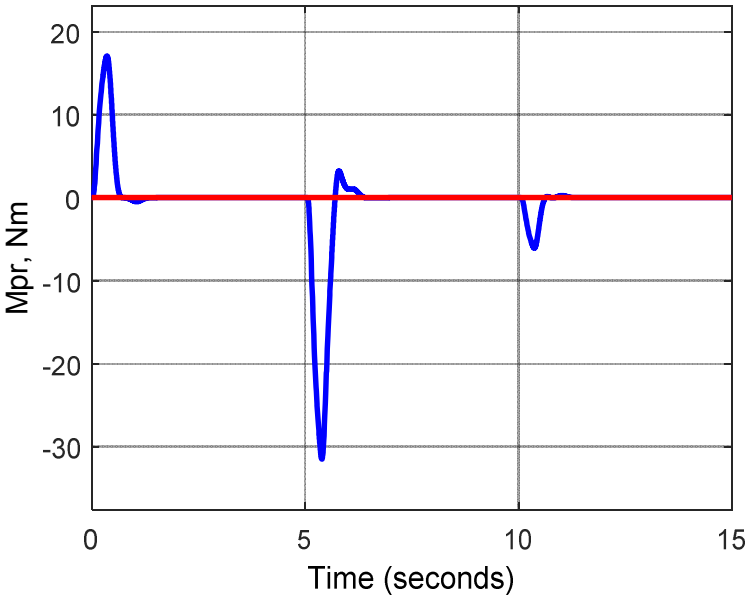


Рисунок 4.27 – Імітаційна мдель двомасової системи з нейрорегулятором

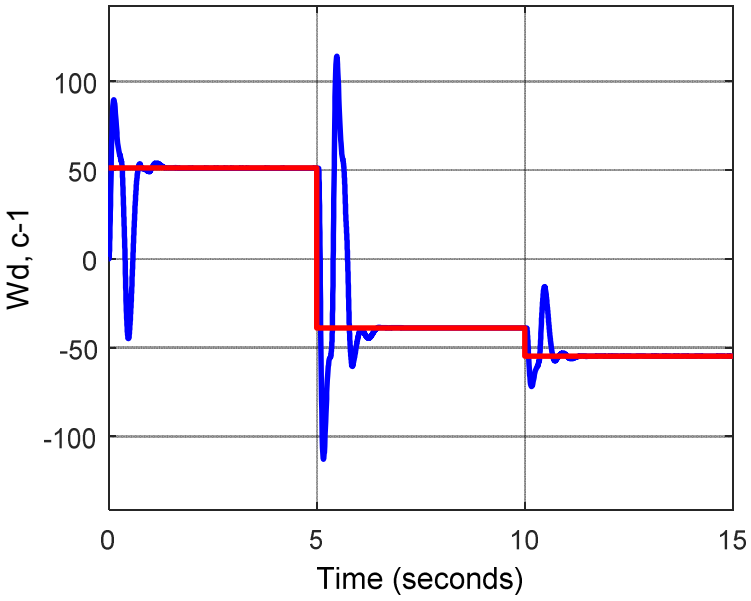


а)

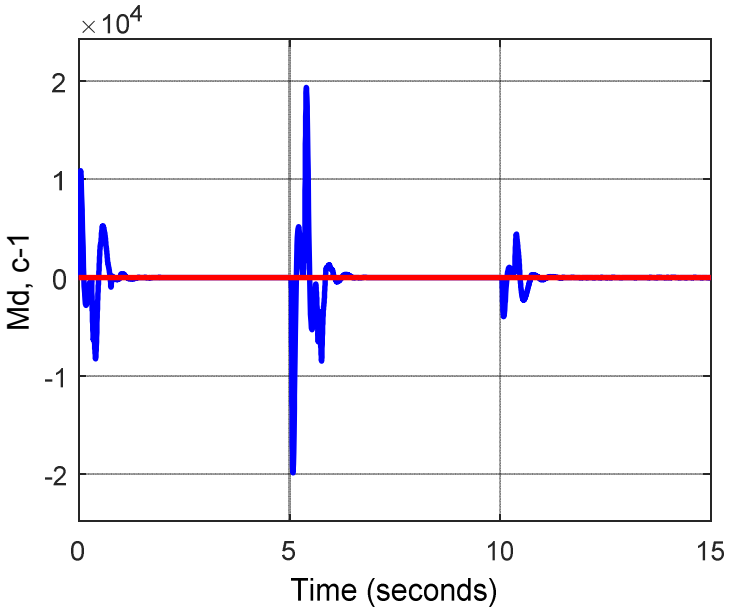


б)

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 $M_{пр} = f(t)$



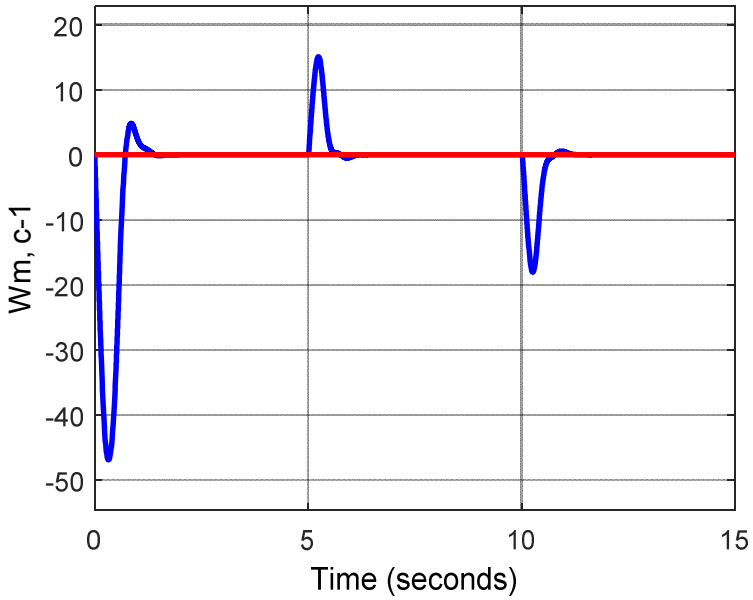
а)



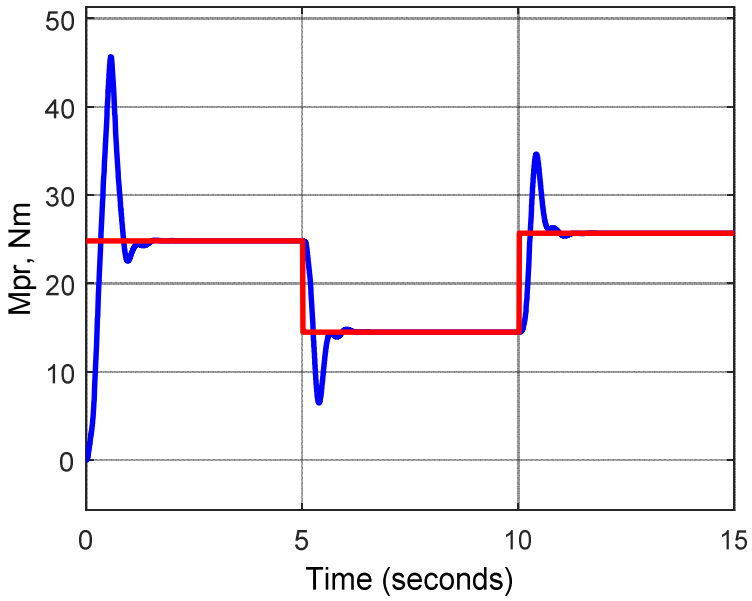
б)

Рисунок 4.29 – Динаміка двигуна двомасової системи з нейрорегулятором під впливом задаючої величини:

а) швидкість обертання двигуна $\omega_д = f(t)$, б) момент двигуна $M_д = f(t)$



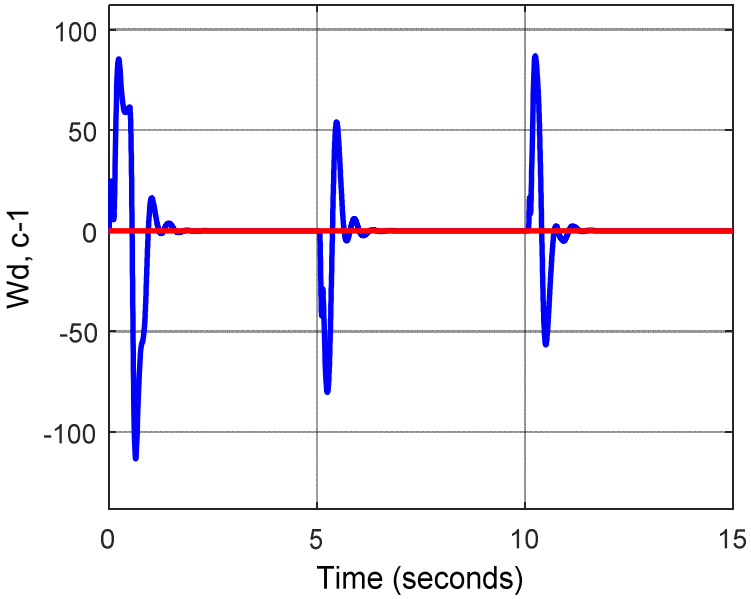
а)



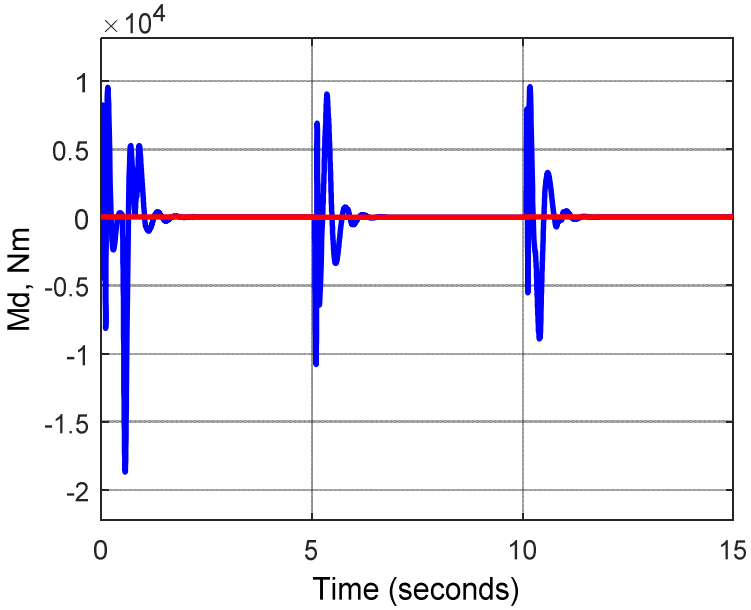
б)

Рисунок 4.30 – Перехідні процеси механічних параметрів двомасової системи з нейрорегулятором під дією зовнішнього збурення:

а) швидкість механізму $\omega_{м} = f(t)$, б) момент пружності $M_{пр} = f(t)$



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. Проаналізовано основні принципи побудови нейрорегуляторів, реалізованих у пакеті Neural Network Toolbox системи MATLAB, зокрема: регулятор із прогнозом (NN Predictive Controller), регулятор на основі моделі авторегресії з ковзним середнім (NARMA-L2 Controller) та регулятор за еталонною моделлю (Model Reference Controller). Для системи управління механізмом підйому мостового крана обрано регулятор із прогнозом NN Predictive Controller. Розглянуто його структуру та основні етапи синтезу.

2. У середовищі SIMULINK системи MATLAB побудовано модель системи управління із застосуванням нейрорегулятора NN Predictive Controller. Система включає блок керованого об'єкта, що представляє двомасову модель механізму підйому мостового крана, та блок регулятора NN Predictive Controller.

3. Виконано генерацію навчальної послідовності для нейромережі шляхом варіювання параметрів та використання моделі виконавчого механізму. Оптимальні параметри навчання визначені наступним чином: довжина навчальної вибірки – 30000 точок, інтервал між вимірами – 0,02 с, мінімальний інтервал ідентифікації – 4 с, максимальний інтервал ідентифікації – 10 с. Це забезпечило високу точність навчання нейромережі.

4. Створено двошарову нейронну мережу з прямою передачею сигналу і проведено її навчання за алгоритмом Левенберга-Марквардта. Визначено оптимальну кількість нейронів у прихованому шарі. Похибка навчання мережі мала порядок 10^{-4}, а миттєві помилки на навчальній, контрольній і тестовій множині не перевищували 0,2. Коефіцієнт кореляції R дорівнював 1, що підтверджує високу точність моделі.

5. Виконано синтез нейрорегулятора NN Predictive Controller із передбаченням на основі параметрів моделі об'єкта управління. Варіювання параметрів нейрорегулятора дозволило визначити ключові, що найбільше впливають на якість регулювання, та встановити їх оптимальні значення. Використання точ-

ної нейромережевої моделі об’єкта забезпечило високі показники точності та якості регулювання.

6. Проведено моделювання роботи системи з синтезованим нейрорегулятором. Аналіз результатів показав, що перехідні процеси змінних стану в режимі пуску та при різкій зміні навантаження мають задовільний характер. Система є астатичною; перерегулювання не перевищує 2%, а час регулювання складає 3 с, що відповідає заданим вимогам.

ВИСНОВКИ

У магістерській кваліфікаційній роботі розв’язано актуальну науково-практичну задачу створення нейромережевої системи управління багатомасовою електромеханічною системою механізму підйому мостового крана, що забезпечує високі показники якості регулювання.

Основні результати роботи можна узагальнити наступним чином:

1. Проведено аналіз систем управління основних механізмів мостового крана з урахуванням сучасних вимог до точності та швидкодії електроприводів. Науково обґрунтовано доцільність застосування нейромережевих технологій для управління багатомасовою електромеханічною системою механізму підйому, що дозволяє підвищити якість регулювання та швидкодію системи.

2. Вибрано електропривод постійного струму за системою ТП-Д як найбільш придатний для керування механізмом підйому. Виконано технічний підбір електродвигуна та силового обладнання електроприводу, проведено перевірку двигуна за умовами нагріву та перевантажувальної здатності. Розрахунки показали відповідність обраного обладнання експлуатаційним вимогам.

3. Розроблено структуру та проведено розрахунок системи автоматичного управління електроприводом. Застосовано двоконтурну систему підлеглого регулювання з внутрішнім контуром струму та зовнішнім контуром ЕРС, що забезпечує необхідну точність управління. Для підвищення якості переходових процесів у контур струму включено внутрішній контур напруги тиристорного перетворювача. Проведено синтез послідовних коректуючих пристроїв: контур струму оптимізовано за модульним критерієм, контур ЕРС – за симетричним. Виконано комп’ютерне моделювання системи у MATLAB, що дозволило оцінити динамічні характеристики в реальному часі.

4. Розроблено математичну модель двомасової системи, яка враховує пружність механічних зв’язків підйомного канату та інших елементів механізму. Аналіз моделювання показав, що основні координати системи демонструють значні коливання під час перехідних процесів, що обумовлено наявністю

пружних механічних зв’язків між масами. Це підкреслює важливість врахування пружності в системах високоточного управління.

5. Для покращення динамічних характеристик та забезпечення необхідної швидкодії розроблено нейромережеву систему управління з регулятором NN Predictive Controller, реалізованим у MATLAB Neural Network Toolbox. Виконано синтез регулятора, описано структуру та порядок навчання нейромережі, проведено генерацію навчальної послідовності з оптимальними параметрами. Розроблено двошарову нейронну мережу з прямою передачею сигналу, проведено навчання алгоритмом Левенберга-Марквардта та визначено оптимальну кількість нейронів у прихованому шарі. Показники точності навчання та кореляції підтвердили високу якість моделі.

6. Проведено моделювання нейромережевої системи управління у середовищі SIMULINK. Аналіз перехідних процесів показав, що система демонструє астатичну поведінку, швидко загасаючі коливання та високу точність регулювання. Перерегулювання не перевищує 2%, час регулювання складає 3 с, що відповідає технічним вимогам до електроприводу підйомного механізму мостового крана. Система ефективно забезпечує стабільність і точність управління при змінних навантаженнях і ступінчастих сигналах на вході.

7. Підсумовуючи, запропонована нейромережева система управління дозволяє суттєво покращити динамічні характеристики багатомасових електромеханічних систем, забезпечує точне та стабільне регулювання швидкості підйому, а також демонструє високу стійкість до коливань навантаження, що підтверджує її практичну придатність для застосування в сучасних мостових кранах.

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. Те-

рейковська, Київ: КПІ ім. Ігоря Сікорського, 2022. - 123 с.

13.Me Culloch W. S., Pitts W. A Logical Calculus of the Ideas Immanent in Nervous Activity // Bulletin of Mathematical Biophysics. – 1943. – № 5. – P. 115-133

14. Hebb D. The Organization of Behavior. – New York: Wiley Publications, 1949.

15.Rosenblatt F. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain // Psychological Review. – 1958. – № 65. – P. 386-408.

16. Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Washington, D.C.: Spartan Books.

17. Hubel, D. H., & Wiesel, T N. (1959). Brain mechanisms of vision. Scientific American, 241, 150-162.

18. Widrow B., Hoff M. E. Adaptive Switching Circuits / IRE WESCON Convention Record – New York, IRE, 1960. – P. 96-104.

19. Widrow B., Winter R. Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition // IEEE Computer. – 1988. – № 21. – № 3. – P. 25-39.

20. Steinbuch, K., & Piske, U. A. W. (1963). Learning matrices and their applications. IRE Transactions on Electronic Computers, EC-12(6), 846–862. .

21. Minsky, M., & Papert, S. (1969). Perceptrons: An Introduction to Computational Geometry. Cambridge, MA: MIT Press.

22. Hopfield J.J. Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons // Proc. of the National Academy of Science. – 1982. – 81. – P. 3088-3092.

23. Hopfield J. J., Tank D. W. Neural Computation of Decisions in Optimization Problems // Biol. Cybernetics. – 1985. – 52. – P. 141-152.

24. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). *Learning representations by back-propagating errors. Nature*, 323(6088), 533–536.

25. Kohonen, T. (1990). *The self-organizing map. Proceedings of the IEEE*, 78(9), 1464–1480

26. 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

27. LeCun, Y., Bengio, Y., & Hinton, G. (2015). *Deep learning. Nature*, 521(7553), 436–444.

28. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* MIT Press.

29. Silver, D., Schrittwieser, J., Simonyan, K., et al. (2017). *Mastering the game of Go without human knowledge. Nature*, 550, 354–359.

30. 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.

31. 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.

32. 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.

33. 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.

34. 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.

35. Danil V. Prokhorov. (2008), Toyota Prius HEV Neurocontrol and Diagnostics, Neural Networks, No. 21, pp. 458-465.

36. 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.

37. 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.

38. 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.
39. Dias F.M., Mota A.M. (2001), Comparison between Different Control Strategies using Neural Networks, 9th Mediterranean Conference on Control and Automation, Croatia, Dubrovnik.
40. 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.
41. 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.
42. 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.
43. Yadav,M.K., Chandra,D. (2015). Керування системами кондиціонування повітря за допомогою нейронної мережі. Міжнародний журнал електроніки та електротехніки, 3(5), 406-410.
44. Камінскі, М. та ін. (2023). Застосування нейронних мереж в електроприводах - тенденції та перспективи. Energies, 16 (11).
45. Jin,L., Li,S., Yu,J., He,J. (2018). Керування роботом-маніпулятором за допомогою нейронних мереж: Огляд. Neurocomputing,285, 23-34.
46. Лі, Т., Чжан, Г., Чжан, Т., Пан, Дж. (2024). Адаптивне нейромережеве відстеження керування роботизованими маніпуляторами на основі спостерігача збурень. Processes, 12(3), 499.
47. 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.

48. 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.

49. Гупта, П., Сінха, Н. К. (2017). Керування роботизованими маніпуляторами за допомогою нейронних мереж: Огляд. У Методах та застосуваннях інтелектуального керування, 103-136. Springer.

50. Astrom K J , Wittenmark B Adaptive control, Reading MA Addison Wesley, 1989.

51. Parks P C Lyapunov redesign of model reference adaptive control systems, IEEE Journal Transactions of Automatic Control, 1966, vol 11, pp 362-367.

52. Hornik K , Stinchcombe M , White H Mylti-layer feed-forward networks are universal approximators Discussion paper, Department of Economics, University of California, San Diego, La Jolla, CA, 1988.

53. Haykin S , Neural Networks A comprehensive foundation, MacMillan College Publishing Co , New York, 1994.

54. Heermann P D Neural network techniques for stable learning control of nonlinear systems Dissertation D S University of Texas at Austin, 1992.

55. Narendra K S , Parthasarathy K Neural networks and dynamical systems Part 1 A gradient approach to Hopfield networks TR 8820, Center for systems science, Department of Electrical Engineering, Yale University, New Haven, CT, 1988.

56. Neural networks for control / Miller W T , Sutton R S , Werbos P J , Eds , The MIT Press, 1990.

57. Werbos P J , Beyond Regression New Tools for Prediction and Analysis in the Behavioral Sciences, Phd Thesis, Dept of Applied Mathematics, Harvard University, Cambridge, Mass , 1974.

58. Ровітхакіс Г. А.: «Stable adaptive neuro-control design via Lyapunov function derivative estimation» // Automatica. – 2001. – Vol. 37, No. 8. – P. 1213–1221. ResearchGate+3ResearchGate+3dl.acm.org+3

59. Tenavi Nakamura-Zimmerer, Qi Gong, Wei Kang: «Neural network optimal feedback control with guaranteed local stability» // arXiv preprint. – 2022. – ID arXiv:2205.00394

60. Gao, Q., Li, H., & Sun, Y. Adaptive neural network-based control for a class of nonlinear pure-feedback systems with time-varying full state constraints // Journal of Applied Science & Engineering. – 2018. – Vol. 11, No. 3. – P. 203-218.

61. Park, J. W., Harley, R. G., Venayagamoorthy, G. K. Indirect adaptive control for synchronous generator: Comparison of MLP/RBF neural networks approach with Lyapunov stability analysis // IEEE Trans. Neural Networks. – 2004. – Vol. 15, No. 2. – P. 460-464. Yonsei University

62. Xiong, Z., Li, J., & Sun, C. Fault-tolerant control for nonlinear systems using sliding mode and adaptive neural network estimator // Soft Computing. – 2020. – Vol. 24. – P. 11535-11544. SpringerLink

63. Fu, X., Li, S., Wunsch, D. C., Alonso, E. Local stability and convergence analysis of neural network controllers with error integral inputs // Elec. & Comp. Engineering Res. & Creative Works. – 2018. – P. (випуск)-(сторінки). Scholars' Mine

64. Garcia C.E. Model predictive control: theory and practice a survey / Garcia C.E., Prett D.M., Morari M. // Automatica. – 1989. – Vol.25. – P.335-348.

65. 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. Розглядається MPC із використанням рекурентної нейронної мережі (RNN) для ідентифікації динаміки системи у фармацевтичному безперервному виробництві. MDPI

66. 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. Застосування нейронної мережі для моделювання нелінійної динаміки та інтеграція з MPC-підходом. NeurIPS Papers

67. Review on model predictive control: an engineering perspective – *The International Journal of Advanced Manufacturing Technology*, 2021, Vol. 117, pp. 1327-1349. Оглядова стаття, яка досліджує MPC – теорію, практику, стійкість, обмеження та впровадження. SpringerLink

68. Efficient model predictive control for nonlinear systems modelled by deep neural networks – 2024, e-print arXiv/preprint. Розробка MPC для нелінійних систем,

моделюваних глибокими нейронними мережами; увага до оптимізації й обмежень вводу/стану.

69. Neural networks for control / eds. W. T. Miller, R. S. Sutton, P. J. Werbos. – Cambridge, MA : The MIT Press, 1990. – 537 p.

70. Solowey D. Neural Generalized Predictive Control / Solowey D., Haley P.J //Proceedingof the 1996 IEEE Symposium on Intelligent Control. – 1996. – P. 277–282.

71. Clarke D.W., Mohtadi C., Tuffs C. Generalized Predictive Control - Part 1: The Basic Algorithm // Automática. – 1987. – Vol. 23. – P. 137-148.

72. Clarke D. W., Mohtadi C., Tuffs C. Generalized Predictive Control. Part 2: The Basic Algorithm // Automática. – 1987. – Vol. 23. – P. 149-163.

73. Clarke D. W. Advances in model-based predictive control.- In Advances in Model-Based Predictive Controled / D. W. Clarke: Oxford University Press. – 1994. – 274

74. Montague G. A., Willis M. J., Tham M. T., Morris A. J. Artificial Neural Network Based Control // Int. Conference on Control. – 1991. – Vol.1. – P. 266-271.

75. Takahashi Y. Adaptive Predictive Control of Nonlinear Time-Varying System using Neural Network //IEEE International Conference on Neural Networks, 1993. – Vol.3. – P.1464-1468

76. Jeong Jun Song, Sunwon Park. Neural Model-Predictive Control for Nonlinear Chemical Processes // Journal of Chemical Engineering of Japan. – 1993. – Vol.26. – №4. –P.347-354.

77. Psichogios D. C., Ungar L. H. Nonlinear Internal Model Control and Model Predictive Control using Neural Networks // 5th IEEE Int. Symposium on Intelligent Control. – 1990. – P.1082-1087.

78. Press W. H., Flannery B. P., Teukolsky S. A., Vetterling V. T. Numerical Recipes.- The Art of Scientific Computing. Cambridge University Press, 1988. – 452.

79. Rumelhart D. E., McClelland J. L. and the PDP Research Group. Parallel Distributed Processing. The MIT Press. – 1986. – Vol.1. – Chapter 8. – 337 p.

80. Теорія автоматичного керування : навчальний посібник / П. В. Леонтьєв

та ін. ; за заг. ред. П. В. Леонтьєва. – Суми : Сумський державний університет, 2024. – 296 с.

81. 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.

82. 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.

83. Теорія автоматичного керування : навчальний посібник / П. В. Леонтьєв та ін. ; за заг. ред. П. В. Леонтьєва. – Суми : Сумський державний університет, 2024. – 296 с.

84. Теорія автоматичного керування : навчальний посібник / О. К. Аблесімов – К. : «Освіта України», 2019. – 270 с.

85. 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.

86. Теорія автоматичного управління : курс лекцій / Т. М. Боровська. – Вінниця : ВНТУ, 2018. – 256 с.

87. 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

88. 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.

89. Автоматизований електропривод. Розділ 1. Теорія електропривода. Навчальний посібник. / Н.Д. Красношапка – К.: КПІ ім. Ігоря Сікорського. - 2023. – 101 с.

90. Возняк, О.М., Штуць. А.А., Колісник М.А. Сучасні системи електроприводів. Теорія та практика. Частина 1. Навчальний посібник. – Вінниця: ТВОРИ, 2021. – 280 с.

91. А. А. Видмиш, Л. В. Ярошенко. Основи електропривода. Теорія та практика. Частина 1. Навчальний посібник. – Вінниця: ВНАУ, 2020. – 387 с.

92. 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.

93. Колб Ант. А, Колб А. А.Теорія електроприводу: Навчальний посібник. – 2-е вид. перероб. і доп. – Д., Національний гірничий університет, 2011. – 540 с.

94. 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.

95. Василега П. О. Електропривод робочих машин : підручник / П. О. Василега. – Суми: Сумський державний університет, 2022. – 290 с. ISBN 978-966-657-900-6.

96. Шефер О.В. Автоматизований електропривод загальнопромислових механізмів: конспект лекцій. – Полтава: ПолтНТУ, 2020. – 154 с.

97. 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.

98.Гурко О.Г. Аналіз та синтез систем автоматичного управління у MATLAB: Навчальний посібник / О.Г. Гурко, І.Ф. Єрьоменко. Харків, ХНАДУ, 2012. – 284 с.

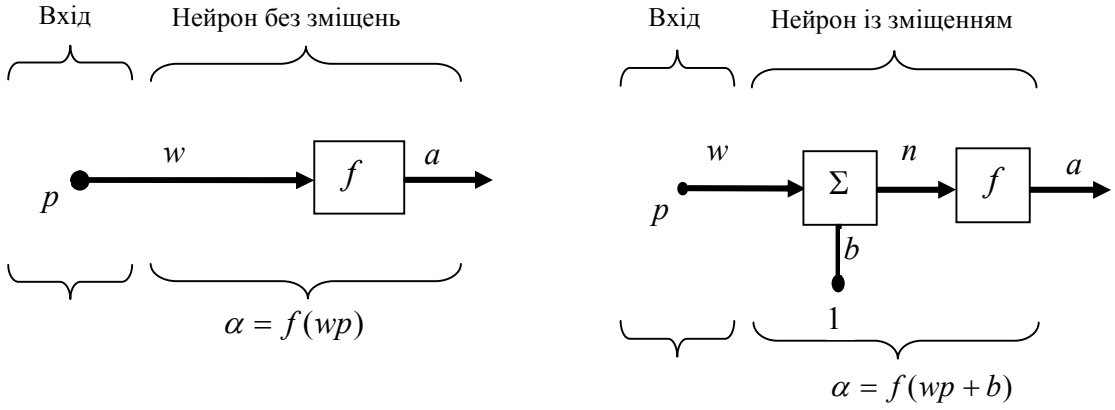
99. Моделювання систем управління в SIMULINK : навч. посібник / В. О. Богомолов, О. Г. Гурко, В. І. Клименко, Д. М. Леонтьєв, О. М. Красюк. Харків ХНАДУ, 2018. – 220 с.

ДОДАТОК А
МОДЕЛІ НЕЙРОНА

А.1 Структурна схема нейрона

Розглянемо основні положення теорії нейронних, які служать базою для розуміння виконаних в роботі досліджень.

Елементарним осередком нейронної мережі є нейрон. Структура нейрона з єдиним скалярним входом показана на рис. А.1, а



а) б)

Рисунок А.1 – Структура простого нейрона:
а) нейрон з одним скалярним входом; б) нейрон з скалярним зсувом

Скалярний вхідний сигнал p умножається на скалярний ваговий коефіцієнт w і результуючий зважений вхід $w \cdot p$ є аргументом функції активації нейрона, яка породжує скалярний вихід a.

Нейрон, показаний на рис. А.1 *б*, доповнений скалярним зсувом b. Зсув підсумовується із зваженим входом $w \cdot p$ і приводить до зрушення аргументу функції f на величину b. Дію зсуву можна звести до схеми зважування, якщо уявити, що нейрон має другий вхідний сигнал із значенням, рівним 1. Вхід n функції активації нейрона як і раніше залишається скалярним і рівним сумі зваженого входу і зсуву b. Ця сума є аргументом функції активації f; виходом

функції активації є сигнал a. Константи w і b є скалярними параметрами нейрона. Основний принцип роботи нейронної мережі полягає в настройці параметрів нейрона так, щоб поведінка мережі відповідала деякій бажаній поведінці. Регулюючи ваги або параметри зсуву, можна навчити мережу виконувати конкретну роботу; можливо також, що мережа сама коректуватиме свої параметри, щоб досягти необхідного результату.

Рівняння нейрона із зсувом має вигляд

$$a = f(w \cdot p + b \cdot 1). \qquad \text{(A.1)}$$

Як вже наголошувалося, зсув b – скалярний параметр нейрона, який не є входом, що настроюється, а константа 1, яка управляє зсувом, розглядається, як вхід і може бути врахована у вигляді лінійної комбінації векторів входу

$$a = \begin{bmatrix} w & p \end{bmatrix} \begin{bmatrix} b \\ 1 \end{bmatrix}. \qquad \text{(A.2)}$$

А.2 Функції активації

Функції активації (передавальні функції) нейрона можуть мати самий різний вигляд. Функція активації f, як правило, належить до класу сигмоїдальних функцій (сигмоїдальною (S-подібною) функцією називається безперервна функція, що має дві горизонтальні асимптоти і одну точка перегину) з аргументом n і виходом a.

Розглянемо три найбільш поширені форми функції активації. Ці функції входять до складу пакету прикладних програм (ППП) Neural Network Toolbox системи MATLAB у вигляді М-функція hardlim, purelin і logsig.

Одинична функція активації з жорстким обмеженням hardlim. Ця функція описується співвідношенням $a = \text{hard}\lim(n) = 1(n)$ і показана на рис.А.2. Вона рівна 0, якщо n <0, і 1, якщо $n \geq 0$.

Лінійна функція активації purelin. Ця функція описується співвідношення $a = \text{purelin}(n) = n$ і показана на рис. А.3.

Логістична функція активації logsig. Ця функція описується співвідношення $a = \text{logsig(n)} = 1/(1+\exp(-n))$ і показана на рис. А.4. Вона належить до класу сигмоїдальних функцій, і її аргумент може приймати будь-яке значення в діапазоні від $-\infty$ до $+\infty$ а вихід змінюється в діапазоні від 0 до 1. Завдяки властивості диференцируемости ця функція часто використовується в мережах з навчанням на основі методу зворотного розповсюдження помилки.

У багатошарових мережах часто застосовуються нелінійні сигмоідальні функції активації типу гіперболічного тангенса (рис. А.5) і функції типу ReLU (рис. А.6)

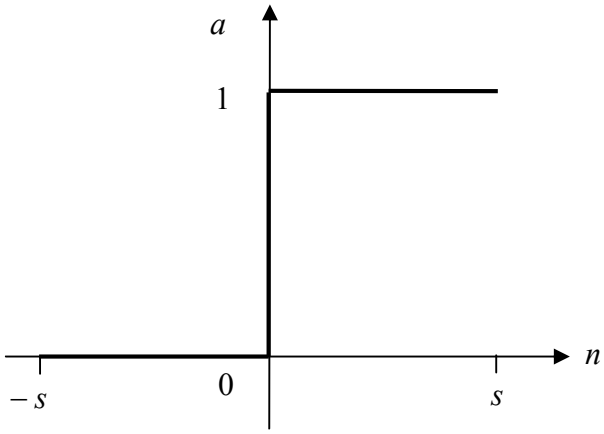


Рисунок А.2 – Одинична Функція активації

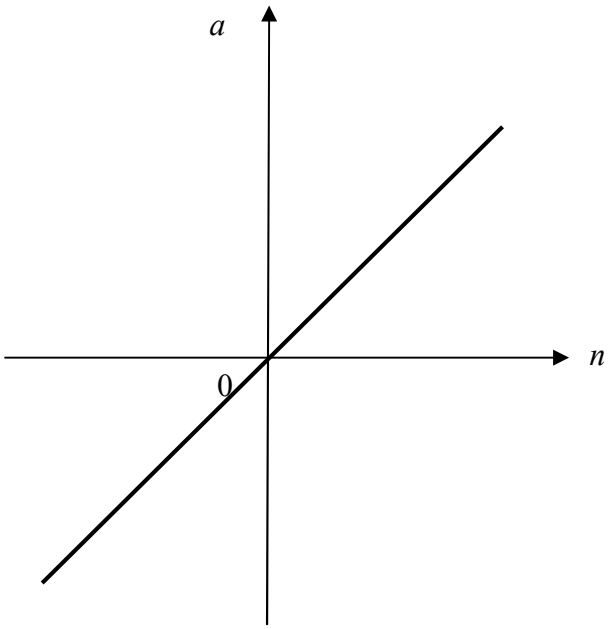


Рисунок А.3 – Лінійна Функція активації

У ППП Neural Network Toolbox включені і інші функції активації. Використовуючи мову MATLAB, користувач може створювати і свої власні унікальні функції.

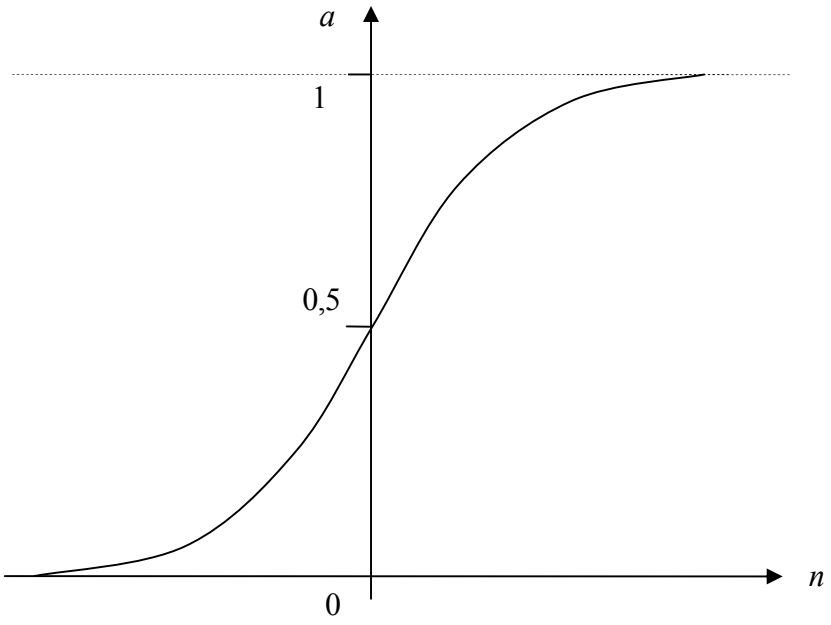


Рисунок А.4 – Логістична функція активації

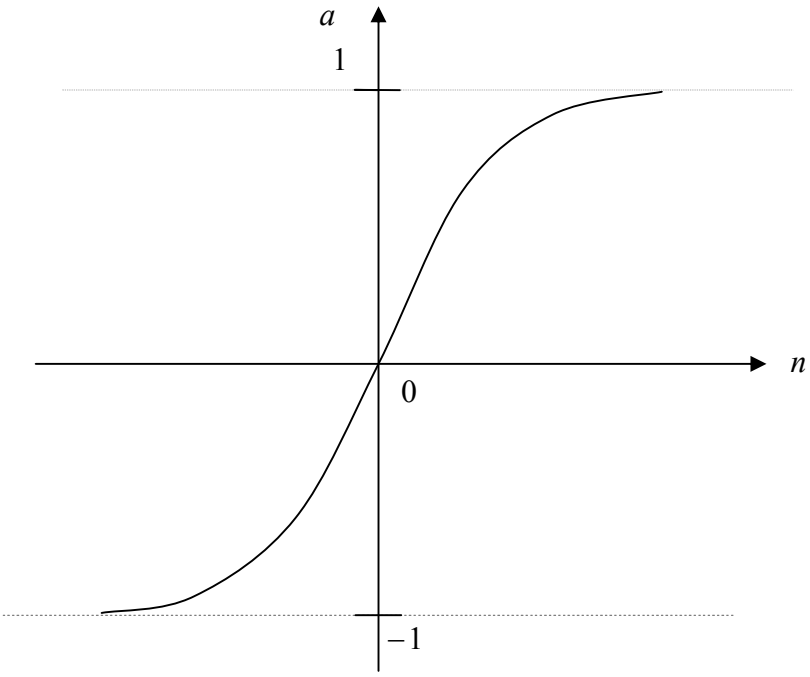


Рисунок А.5 – Функція активації типу гіперболічного тангенса

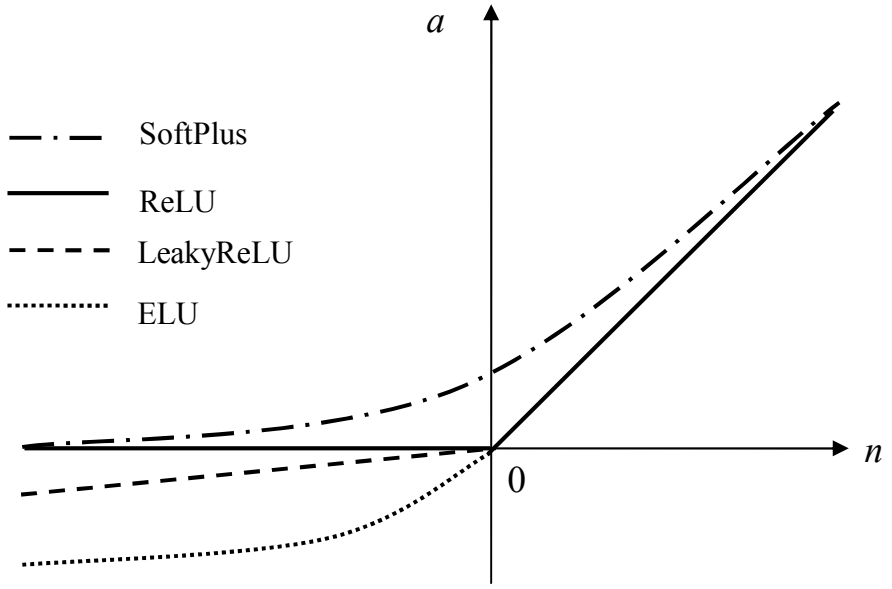


Рисунок А.6 – Функція активації типу ReLU

А.3 Нейрон з векторним входом

Нейрон з одним вектором входу **p** з R елементами $p_1, p_2,..., p_R$ показаний на рис. А.6. Тут кожен елемент входу умножається на ваги $w_{11}, w_{12},..., w_{1R}$ відповідно і зважені значення передаються на суматор. Їх сума рівна скалярному добутку вектора-рядка **W** на вектор входу **p.**

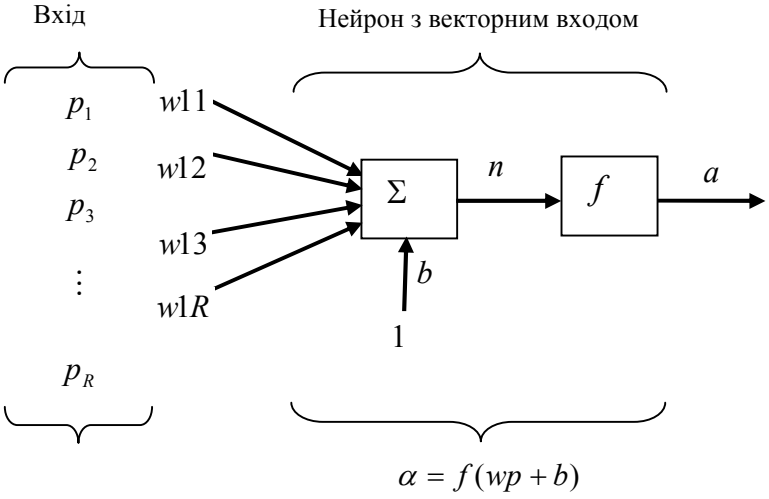


Рисунок А.6 – Схема нейрона з векторним входом

Нейрон має зсув b, який підсумовується із зваженою сумою входів. Результуюча сума n рівна

$$n = w_{11} \cdot p_1 + w_{12} \cdot p_2 + ... + w_{1R} \cdot p_R + b \qquad \text{(A.3)}$$

і служить аргументом функції активації f. У нотації мови MATLAB цей вираз записується так:

$$n = \mathbf{W} \cdot \mathbf{p} + b \ . \qquad \text{(A.4)}$$

Структура нейрона, показана вище, містить багато зайвих деталей. При розгляді мереж, що складаються з великого числа нейронів, в ППП Neural Network Toolbox використовується укрупнена структурна схема нейрона (рис.А.7).

Вхід нейрона зображується у вигляді темної вертикальної межі, під якою указується кількість елементів входу R. Розмір вектора входу **p** указується нижче за символ **p** і рівний R. Вектор входу умножається на вектор-рядок **W** довжини R. Як і раніше, константа 1 розглядається як вхід, який умножається на скалярний зсув b. Входом n функції активації нейрона служить сума зсуву b і добутки $\mathbf{W} \cdot \mathbf{p}$. Ця сума перетвориться функцією активації f на виході якої отримуємо вихід нейрона a, який в даному випадку є скалярною величиною. Структурна схема, приведена на рис. А.7, називається шаром мережі. Шар характеризується матрицею вагів **W**, зсувом b, операціями множення $\mathbf{W} \cdot \mathbf{p}$, підсумовування і функцією активації f. Вектор входів **p** зазвичай не включається в характеристики шару.

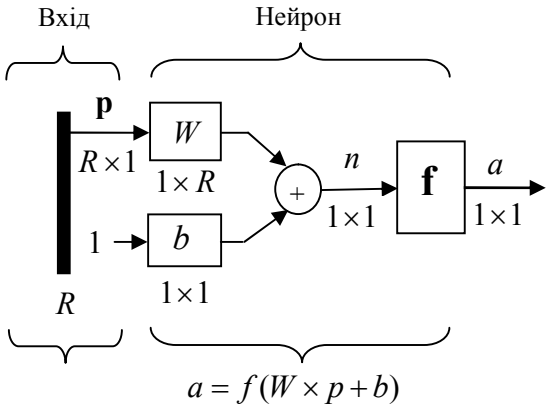


Рисунок А.7 – Укрупнена структурна схема нейрона

ДОДАТОК Б
АРХІТЕКТУРА НЕЙРОННИХ МЕРЕЖ

Б.1 Одношарові мережі

Реальна нейронна мережа може містити один або більша кількість шарів і відповідно характеризуватися як одношарова або як багатошарова.

Розгорнена схема мережі з одного шару з R вхідними елементами і S нейронами показана на рис. Б.1. У цій мережі кожен елемент вектора входу сполучений зі всіма входами нейрона і це з'єднання задається матрицею вагів $\mathbf{W}$; 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 $\mathbf{n}$ функції активації шаруючи. Виходи шару нейронів формують вектор-стовпець $\mathbf{a}$, і, таким чином, опис шару нейронів має вигляд

$$\mathbf{a} = f(\mathbf{W} \cdot \mathbf{p} + \mathbf{b}). \qquad (Б.1)$$

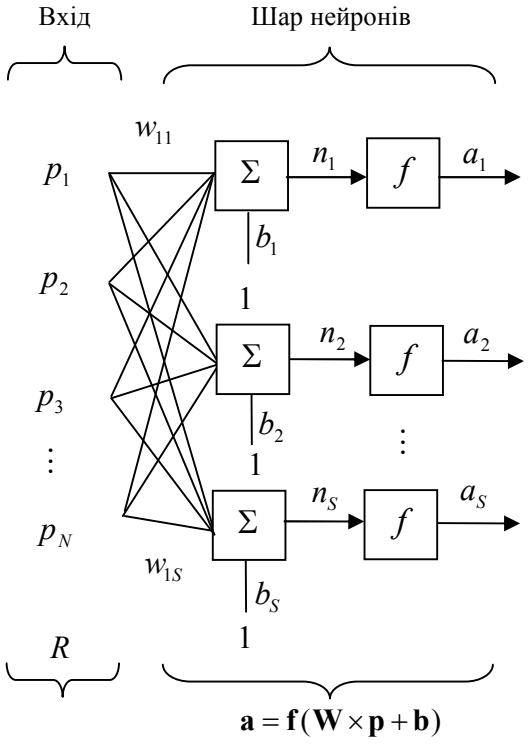


Рисунок Б.1 – Схема одношарової мережі

Кількість входів R в шарі може не співпадати з кількістю нейронів S. 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 зображеній на рис. Б.1, паралельно. Обидві мережі матимуть ті ж самі входи, і кожна мережа генеруватиме певну частину виходів. Елементи вектора входу передаються в мережу через матрицю вагів $\mathbf{W}$, що має вигляд

$$\mathbf{W} = \begin{vmatrix} w_{11} & w_{12} & \dots & w_{1R} \\ w_{21} & w_{22} & \dots & w_{2R} \\ \dots & \dots & \dots & \dots \\ w_{S1} & w_{S2} & \dots & w_{SR} \end{vmatrix}. \qquad \text{(Б.2)}$$

Відмітимо, що індекси рядків матриці $\mathbf{W}$ указують адресатів (пункти призначення) вагів нейронів, а індекси стовпців – яке джерело є входом для цієї ваги. Таким чином, елемент матриці вагів $w_{12} = \mathbf{W}(1,2)$ визначає коефіцієнт, на який умножається другий елемент входу при передачі його на перший нейрон.

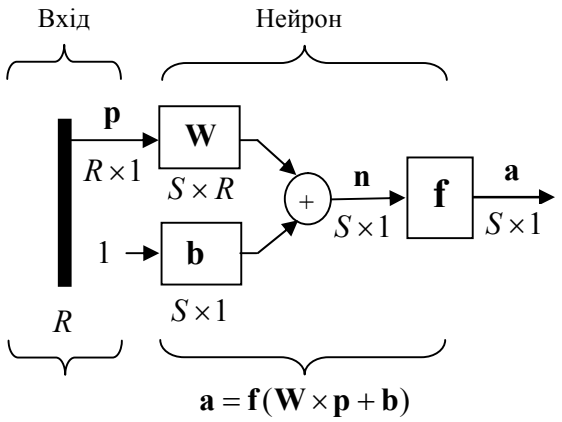


Рисунок Б.2 – Укрупнена структурна схема одношарової мережі

Для одношарової мережі з S нейронами укрупнена структурна схема показана рис. Б.2.

На рисунку позначено: $\mathbf{p}$ – вектор входу розміру $R \times 1$; $\mathbf{W}$ – вагова матриця розміру $S \times R$; $\mathbf{a}$, $\mathbf{b}$, $\mathbf{n}$ – вектори розміру $S \times 1$.

Б.2 Багатошарові мережі

Розглянемо мережі, що мають декілька шарів. Вагові матриці, сполучені з входами, називаються вагами вхідного шару, а вагові матриці для сигналів, витікаючі з шару, називаються вагами вихідного шару. Далі,будемо використовувати верхні індекси, щоб вказати джерело і адресат для різних вагів і інших елементів нейронної мережі. Щоб пояснити це, розглянемо спочатку тільки один, перший шар багатошарової мережі (рис. Б.3)

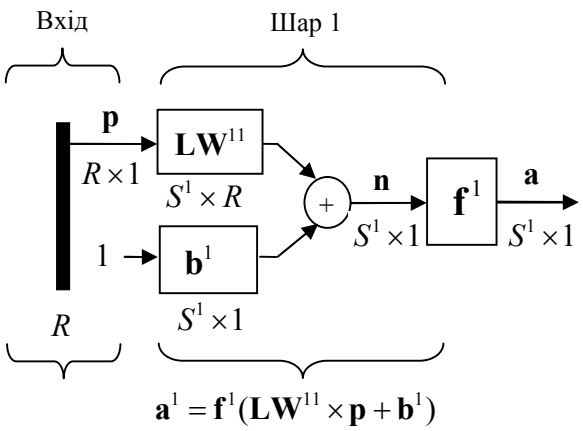


Рисунок Б.3 – Структурна схема першого шару багатошарової мережі

Позначимо вагову матрицю, пов'язану з входами, через $\mathbf{IW}^{11}$, верхні індекси якої указують, що джерелом входів є перший шар (другий індекс) і адреси том є також перший шар (перший індекс). Елементи цього шару, такі, як зсув $\mathbf{b}^1$, вхід функції активації $\mathbf{n}^1$ і вихід шару $\mathbf{a}^1$ мають верхній індекс 1, щоб позначити, що вони пов'язані з першим шаром. Надалі для матриць вагів входу і виходу шару будуть використані позначення **IW** (Input Weight) і **LW** (Layer Weight) відповідно.

Коли мережа має декілька шарів, то кожен шар має свою матрицю вагів $\mathbf{W}$, вектор зсуву $\mathbf{b}$ і вектор виходу $\mathbf{a}$. Щоб розрізняти вагові матриці, вектори виходу і т.д. для кожного з цих шарів, введемо номер шару як верхній індекс для тієї, що представляє інтерес, змінної. Використання цієї системи позначень для мережі з трьох шарів можна бачити на показаній нижче структурній схемі і з рівнянь, приведених в нижній частині рис. Б.4.

Мережа, показана вище, має R входів, S^1 нейронів в першому шарі, S^2 нейронів в другому шарі і т.д. Для спільності вважатимемо, що різні шари мають різне число нейронів. На зсуви для кожного нейрона поданий постійний вхідний сигнал 1. Відмітимо, що виходи кожного проміжного шару служать

входами для наступного шару. Таким чином, шар 2 може бути розглянутий як один шар мережі з S^1 входами S^2 нейронами і $S^1 \times S^2$ матрицею вагів $\mathbf{W}_2$. Вхід до шару 2 є 1, а вихід – 2. Тепер, коли позначені всі вектори і матриці шару 2, можна трактувати його як самостійну одношарову мережу. Такий підхід може бути використаний до будь-якого шару мережі.

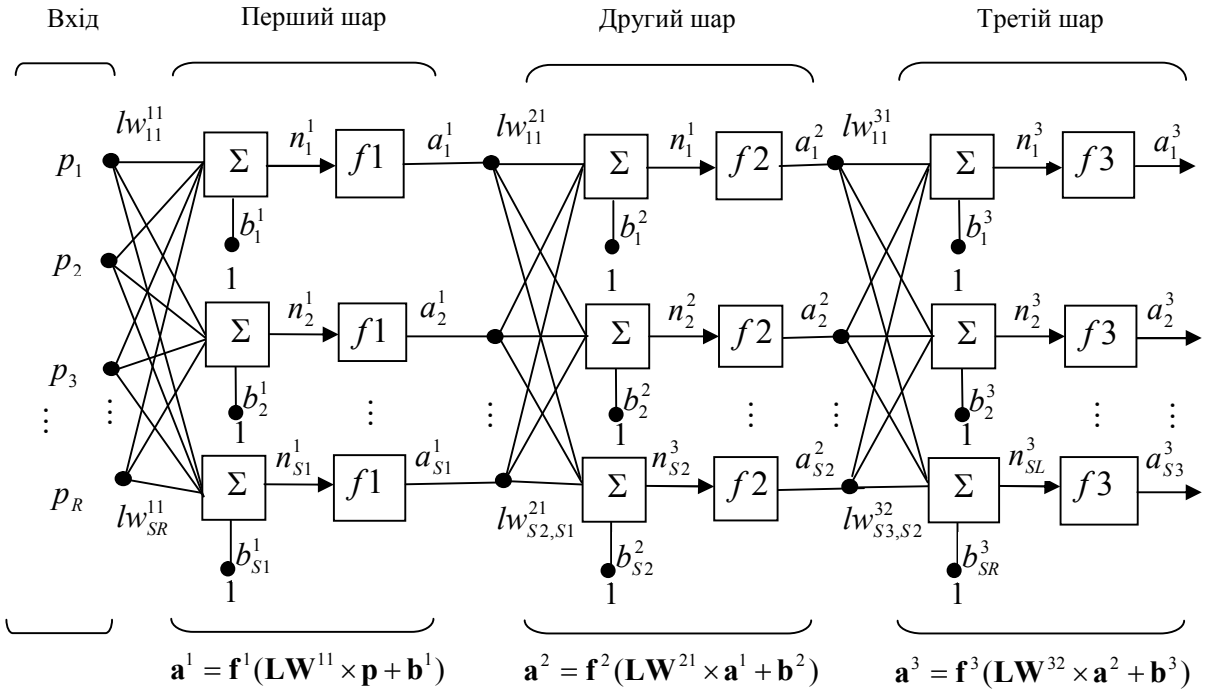


Рисунок Б.4 – Структурна схема багатошарової мережі

Шари багатошарової мережі мають різні призначення. Шар, який утворює вихід мережі, називається шаром виходу. Всі інші шари називаються прихованими шарами. Тришарова мережа, показана вище, має вихідний шар (шар 3) і 2 прихованих шари (шар 1 і шар 2). Ця ж тришарова мережа може бути представлена у вигляді укрупненої структурної схеми (рис. Б.5).

Відмітимо, що вихід третього шару $\mathbf{a}^3$ позначений через **y**. Ця зроблено для того, щоб підкреслити, що вихід останнього шару є виходом мережі.

Багатошарові мережі володіють вельми потужними можливостями. Наприклад, двошарова мережа, в якій перший шар містить сигмоїдальну, а другий шар – лінійну функцію активації, може бути навчена апроксимувати з довільною точністю будь-яку функцію з кінцевим числом точок розриву.

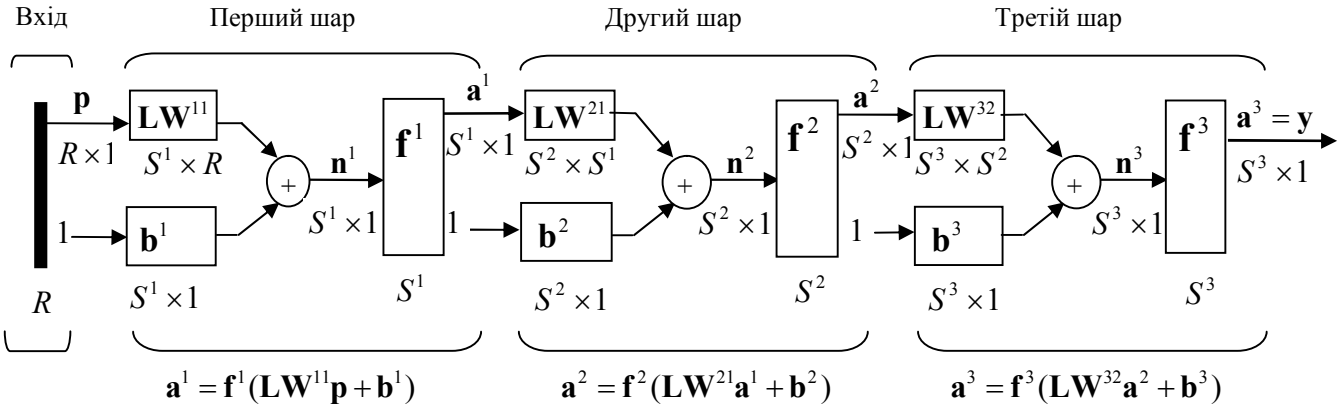


Рисунок Б.5 – Укрупнена структурна схема багатошарової мережі

На закінчення можна сформулювати наступні висновки. Вхід функції активації нейрона визначається зсувом і сумою зважених входів. Вихід нейрона залежить як від входів нейрона, так і від виду функції активації. Один нейрон не може вирішувати складні завдання, проте декілька нейронів, об'єднаних в один або декілька шарів, володіють великими можливостями.

Архітектура мережі складається з опису того, скільки шарів має мережу, кількості нейронів в кожному шарі, виду функції активації кожного шару і інформації про з'єднання шарів. Архітектура мережі залежить від того конкретного завдання, яке повинна вирішувати мережа.

Робота мережі полягає в обчисленні виходів мережі на основі відомих входів з метою формування бажаного відображення вхід/вихід. Конкретне завдання визначає число входів і число виходів мережі. Окрім числа нейронів у вихідному шарі мережі, для проектувальника важливе число нейронів в кожному шарі. Більша кількість нейронів в прихованих шарах забезпечує могутнішу мережу. Якщо повинне бути реалізоване лінійне відображення, то слід викори-

стовувати нейрони з лінійними функціями активації. При цьому треба пам'ятати, що лінійні нейронні мережі не можуть формувати нелінійні відображення. Використання нелінійних функцій активації дозволяє набудувати нейронну мережу на реалізацію нелінійних зв'язків між входом і виходом.

Мережі із зсувом дозволяють формувати складніші зв'язки між входами і виходами, чим мережі без зсуву. Наприклад, нейрон без зсуву, коли всі входи нульові, завжди задаватиме вхід функції активації рівним нулю, проте нейрон із зсувом може бути навчений так, щоб за тих же умов задати вхід функції активації довільної форми.

У багатошарових мережах часто застосовуються нелінійні сигмоїдальні функції активації типа логістичної (див. рис. А.4) або гіперболічного тангенса (див. рис. А.5).

Якщо останній шар багатошарової мережі використовує такі функції активації, то виходи мережі будуть обмежені. Коли у вихідному шарі використовуються лінійні нейрони, то виходи мережі можуть приймати довільні значення. У ППП NNT передбачені М-функції, що дозволяють обчислювати похідні функцій активації.

Б.3.Мережі з прямою передачею сигналу

Одношарова мережа з S нейронами з функціями активації logsig, що має R входів, показана на рис. Б.6. Ця мережа, що не має зворотних зв'язків, називається мережею з прямою передачею сигналу. Такі мережі часто мають один або більше прихованих шарів нейронів з сигмоїдальними функціями активації, тоді як вихідний шар містить нейрони з лінійними функціями активації. Мережі з такою архітектурою можуть відтворювати вельми складні нелінійні залежності між входом і виходом мережі.

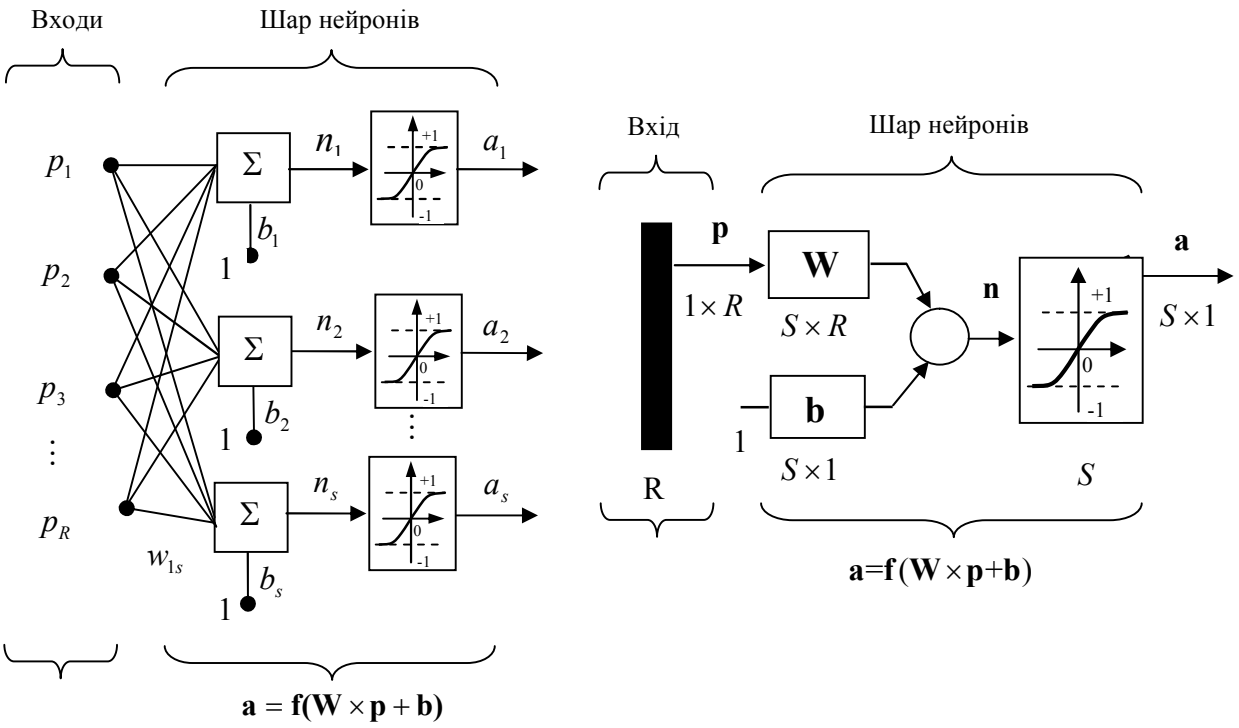


Рисунок Б.6 –Структурная схема одношарової мережі з R входами

Двошарова мережа, показана на рис. Б.7, може бути використана для апроксимації функцій. Вона може достатньо точно відтворити будь-яку функцію з кінцевим числом точок розриву, якщо задати достатнє число нейронів прихованого шару.

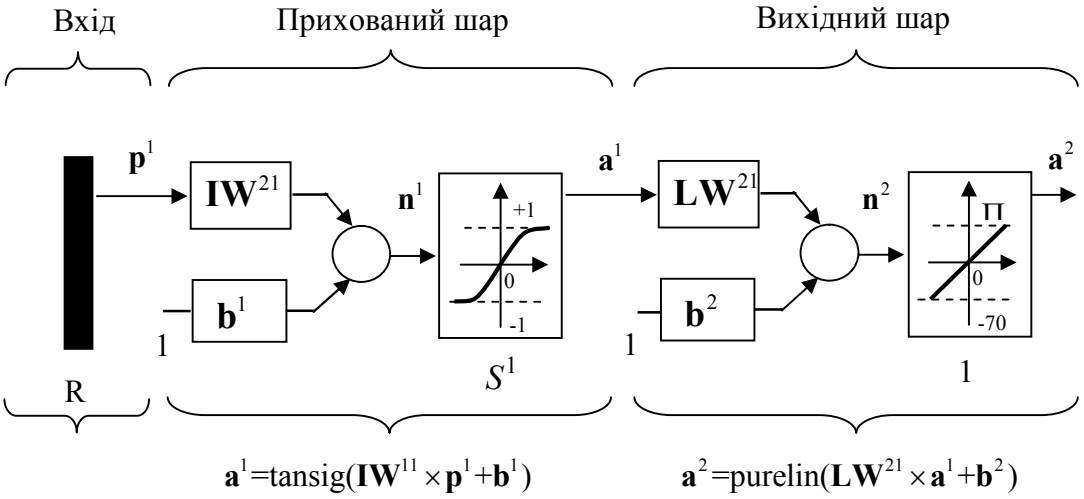


Рисунок Б.7 – Структурна схема багатошарової мережі з R входам

ДОДАТОК В
НАВЧАННЯ НЕЙРОННИХ МЕРЕЖ

В.1 Сутність процесу навчання

Доведено, що будь-яка безперервна функція на замкнутій обмеженій множині може бути рівномірно наближена функціями, обчислюваними нейронними мережами, якщо функція активації нейрона двічі безперервно диференцируєма і нелінійна (існують і інші варіанти теорем про облиште).

Таким чином, нейронні мережі є універсальними апроксимуючими системами.

Очевидно, що процес функціонування ШНМ, тобто суть дій, які вона здатна виконувати, залежить від величин синаптичних зв'язків, тому, задавшись певною структурою ШНМ, що відповідає якому-небудь завданню, розробник мережі повинен знайти оптимальні значення всіх вагових коефіцієнтів.

Цей етап називається навчанням ШНМ, і від того, наскільки якісно він буде виконаний, залежить здатність мережі вирішувати поставлені перед нею проблеми під час функціонування.

Навчання штучною нейронною полягає в настройці вагових коефіцієнтів $w_{i,j}$ її базових процесорних елементів, результатом чого є виконання мережею конкретних завдань – розпізнавання, оптимізації, апроксимації, управління. Досягнення подібної мети формалізується критерієм якості *Q,* мінімальне значення $\min_w Q = Q^*$ якого відповідає якнайкращому рішенню поставленої задачі.

Навчання мережі відбувається відповідно до схеми, показаної на рис.В.1.

При навчанні мережі є деяка база даних, що містить приклади. Пред'являючи перший приклад на вхід мережі, отримуємо від неї деяку відповідь, не обов'язково вірну. Відома і вірна (бажаний) відповідь. Обчислюючи різницю між бажаною відповіддю і реальною відповіддю мережі, отримуємо вектор помилки. Алгоритм навчання – це набір формул, який дозволяє по вектору поми-

лки обчислити необхідні поправки для вагів мережі. Один і той же приклад може пред'являтися мережі багато раз.

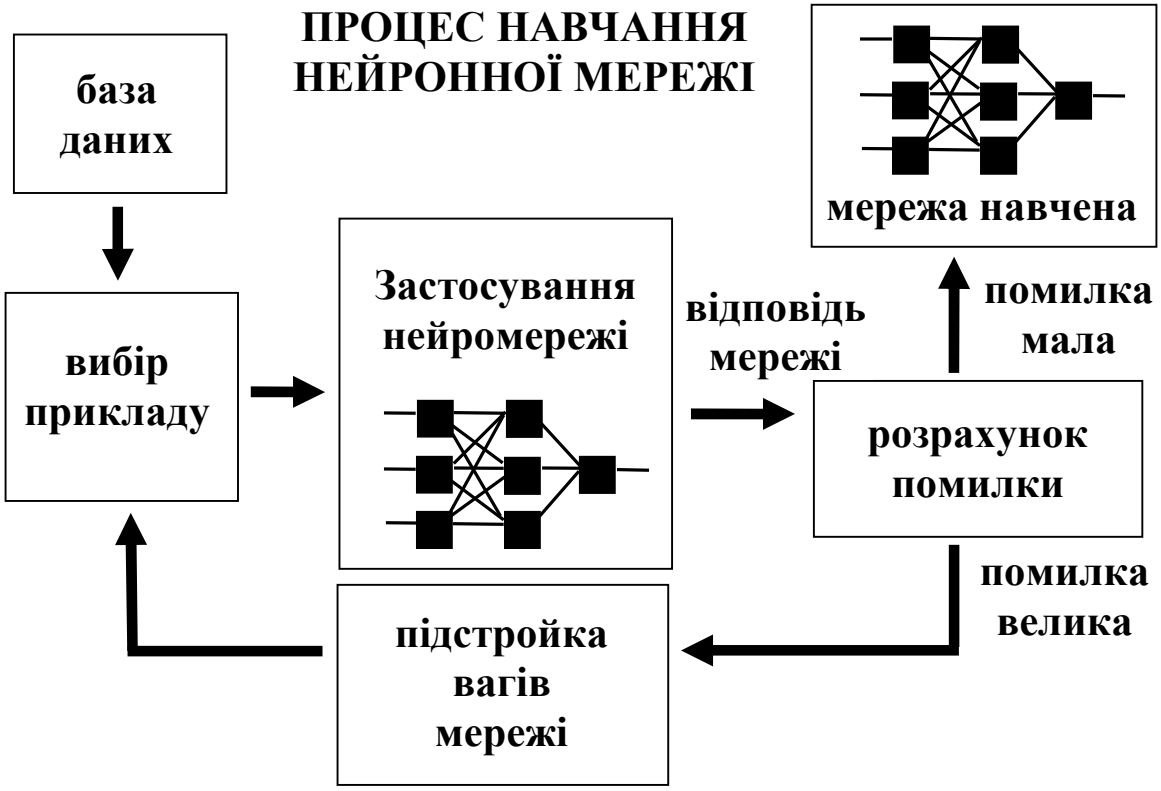


Рисунок В.1 – Ілюстрація процесу навчання НМ

Виявляється, що після багатократного пред'явлення прикладів ваги мережі стабілізуються, причому мережа дає правильні відповіді на всі (або майже все) приклади з бази даних. У такому разі говорять, що «мережа вивчила всі приклади», «мережа навчена», або «мережа натренована». У програмних реалізаціях можна бачити, що в процесі навчання функція помилки (наприклад, сума квадратів помилок по всіх виходах) поступово зменшується. Коли функція помилки досягає нуля або прийнятного малого рівня, тренування зупиняють, а отриману мережу вважають натренованою і готовою до застосування на нових даних.

Важливо відзначити, що вся інформація, яку мережа має про завдання, міститься в наборі прикладів. Тому якість навчання мережі безпосередньо залежить від кількості прикладів в навчальній вибірці, а також від того, наскільки

повно ці приклади описують дане завдання. Так, наприклад, безглуздо використовувати мережу для прогнозу фінансової кризи, якщо в навчальній вибірці криз не представлено. Вважається, що для повноцінного тренування потрібно хоч би декілька десятків (а краще за сотні) прикладів.

Математично процес навчання можна описати таким чином.

В процесі функціонування нейронна мережа формує вихідний сигнал $\mathbf{Y}$ відповідно до вхідного сигналу $\mathbf{X}$, реалізовуючи деяку функцію $\mathbf{Y} = G(\mathbf{X})$. Якщо архітектура мережі задана, то вид функції G визначається значеннями синаптичних вагів і зсувів мережі.

Хай рішенням деякої задачі є функція $\mathbf{Y} = F(\mathbf{X})$, задана парами вхідних, – вихідних даних $(\mathbf{X}^1, \mathbf{Y}^1)$, $(\mathbf{X}^2, \mathbf{Y}^2)$, ..., $(\mathbf{X}^N, \mathbf{Y}^N)$, для яких $\mathbf{Y}^k = F(\mathbf{X}^k))$ $(k = 1...N)$.

Навчання полягає в пошуку (синтезі) функції *G*, близької до F і мінімізації деякої функції помилки E.

Якщо вибрана безліч навчальних прикладів – пара $(\mathbf{X}^k, \mathbf{Y}^k)$ (де $k = 1...N$) і спосіб обчислення функції помилки *E*, те навчання нейронної мережі перетворюється на завдання багатовимірної оптимізації, дуже велику розмірність, що має, при цьому, оскільки функція E може мати довільний вигляд, навчання в загальному випадку – багатоекстремальне неопукле завдання оптимізації.

В.2 Алгоритми навчання нейронних мереж

В.2.1 Класифікація алгоритмів навчання.

Для вирішення цього завдання можуть бути використані наступні (ітераційні) алгоритми.

1 – Алгоритми локальної оптимізації з обчисленням приватних похідних першого порядку:

градієнтний алгоритм (метод швидкого спуску);

методи з одновимірною і двовимірною оптимізацією цільової

функції у напрямі антиградієнта;

метод зв'язаних градієнтів;

методи, що враховують напрям антиградієнта на декількох кроках алгоритму.

2 – Алгоритми локальної оптимізації з обчисленням приватних похідних першого і другого порядку:

метод Ньютона;

методи оптимізації з розрідженими матрицями Гессе;

квазіньютонівські методи;

метод Гаусса-Ньютона;

метод Левенберга-Марквардта і ін.

3 – Стохастичні алгоритми оптимізації:

пошук у випадковому напрямі;

імітація віджигу;

метод Монте-Карло (чисельний метод статистичних;

випробувань).

4) Алгоритми глобальної оптимізації (завдання глобальної оптимізації вирішуються за допомогою перебору значень змінних, від яких залежить цільова функція).

В.2.2 Алгоритм зворотного розповсюдження.

Розглянемо ідею одного з найпоширеніших алгоритмів навчання – алгоритму зворотного розповсюдження помилки (БР) (back propagation). Це ітеративний градієнтний алгоритм навчання, який використовується з метою мінімізації середньоквадратичного відхилення поточного виходу від бажаного виходу в багатошарових нейронних мережах.

Алгоритм зворотного розповсюдження використовується для навчання багатошарових нейронних мереж з послідовними зв'язками. Як зазначено вище, нейрони в таких мережах діляться на групи із загальним вхідним сигналом – шари, при цьому на кожен нейрон першого шару подаються всі елементи зов-

нішнього вхідного сигналу, а всі виходи нейронів q-го шару подаються на кожен нейрон шаруючи ($q+1$). Нейрони виконують зважене (з синаптичними вагами) підсумовування елементів вхідних сигналів; до даної суми додається зсув нейрона. Над отриманим результатом потім виконується нелінійне перетворення за допомогою активаційної функції. Значення функції активації є вихід нейрона.

У багатошарових мережах оптимальні вихідні значення нейронів всіх шарів, окрім останнього, як правило, невідомі, і трьох- або більш шаровий персептрон вже неможливо навчити, керуючись тільки величинами помилок на виходах ШНМ. Найбільш прийнятним варіантом навчання в таких умовах виявився градієнтний метод пошуку мінімуму функції помилки з розглядом сигналів помилки від виходів ШНМ до її входів, тобто в напрямі, зворотному прямому розповсюдженню сигналів в звичайному режимі роботи. Цей алгоритм навчання ШНМ отримав назву процедури зворотного розповсюдження.

У даному алгоритмі функція помилки є сумою квадратів розузгодження (помилки) бажаного виходу мережі і реального. При обчисленні елементів вектора градієнта використаний своєрідний вид похідних функцій активації сигмоїдального типу. Алгоритм діє циклічно (ітеративно), і його цикли прийнято називати епохами. На кожній епосі на вхід мережі по черзі подаються всі навчальні спостереження, вихідні значення мережі порівнюються з цільовими значеннями і обчислюється помилка. Значення помилки, а також градієнта поверхні помилок використовується для коректування вагів, після чого всі дії повторюються. Початкова конфігурація мережі вибирається випадковим чином, і процес навчання припиняється або коли пройдена певна кількість епох, або коли помилка досягне деякого певного рівня малості, або коли помилка перестане зменшуватися (користувач може сам вибрати потрібну умову зупинки).

Приведемо словесний опис алгоритму.

Крок 1. Вагам мережі присвоюються невеликі початкові значення.

Крок 2. Вибирається чергова навчальна пара (***X***, У) з навчальної множини; вектор Х подається на вхід мережі.

Крок 3. Обчислюється вихід мережі.

Крок 4. Обчислюється різниця між необхідним (цільовим, ***Y***) і реальним (обчисленим) виходом мережі.

Крок 5. Ваги мережі коректуються так, щоб мінімізувати помилку (спочатку ваги вихідного шаруючи, потім, з використанням правила диференціювання складної функції і відміченого своєрідного виду похідної сигмоїдальної функції, – ваги попереднього шару і т. п.).

Крок 6. Кроки з 2-го по 5-ій повторюються для кожної пари навчальної множини до тих пір, поки помилка на всій множині не досягне прийнятної величини.

Кроки 2 і 3 подібні до тих, які виконуються у вже навченій мережі.

Обчислення в мережі виконуються пошарово. На кроці 3 кожний з виходів мережі віднімається з відповідного компоненту цільового вектора з метою отримання помилки. Ця помилка використовується на кроці 5 для корекції вагів мережі.

Кроки 2 і 3 можна розглядати як «прохід вперед», оскільки сигнал розповсюджується по мережі від входу до виходу. Кроки 4 і 5 складають «зворотний прохід», оскільки тут обчислюваний сигнал помилки розповсюджується назад по мережі і використовується для підстроювання вагів.

Класичний метод зворотного розповсюдження відноситься до алгоритмів з лінійною збіжністю. Його відомими недоліками є: невисока швидкість збіжності (велике число ітерацій, потрібних для досягнення мінімуму функції помилки), можливість сходитися не до глобального, а до локальних рішень (локальним мінімумам відміченої функції). Можливий також параліч мережі, при якому більшість нейронів функціонують при дуже великих значеннях аргументу функції активації, тобто на її пологій ділянці (оскільки помилка пропорційна похідною, яка на даних ділянках мала, то процес навчання практично завмирає).

Для усунення цих недоліків були запропоновані численні модифікації алгоритму зворотного розповсюдження, які зв'язані з використанням різних фун-

кцій помилки, різних процедур визначення напряму і величини кроку і т.п. Розглянемо один з них.

В.3 Метод зворотного розповсюдження помилки

Як було зазначено вище, метод зворотного розповсюдження помилки – метод ВР розроблений для навчання статичних нелінійних багатошарових нейронних мереж і використовує технологію послідовної і пошарової настройки базових процесорних елементів, починаючи з останнього, вихідного шару і закінчуючи настройкою елементів першого шару. «Урок» навчання штучної нейронної мережі може бути повторений необхідне число разів. Для настройки вагових коефіцієнтів базових елементів застосовується δ-правило в його загальній формі: у вигляді алгоритму градієнтного методу мінімізації критерію навчання $Q(\mathbf{e}_{\text{u}})$. Тут $\mathbf{e}_{\text{u}} = \mathbf{u}^* - \mathbf{u} = \mathbf{u}^* - \mathbf{q}^{(K)}$ – вектор помилки навчання мережі щодо бажаного (еталонного) виходу $\mathbf{u}^*$ при нелінійному перетворенні вхідного вектора $\mathbf{r}$ у вихідний вектор $\mathbf{u} = \mathbf{q}^{(K)}$ останнього шару K. Помилка $\mathbf{e}_{\text{u}}$, явним чином залежить від коефіцієнтів $w_{i,j}^{(K)}$ і може бути використана як аргумент функціонала Q для настройки базових елементів K-го шару. Явної залежності вектора $\mathbf{e}_{\text{u}}$ і функції $Q(\mathbf{e}_{\text{u}})$ від вагових коефіцієнтів $w_{i,j}^{(l)}$ $(l = \overline{1,\ K-1})$ немає. Тому у разі використань критерію навчання $Q(\mathbf{e}_{\text{u}})$ для настройки коефіцієнтів $w_{i,j}^{(l)}$ помилка $\mathbf{e}_{\text{u}}$ в процесі навчання багатошарової нейронної мережі перераховується послідовно в узагальнені помилки $\boldsymbol{\sigma}^{(l)}$, що явно залежать від значень $w_{i,j}^{(l)}$. Настройка коефіцієнтів $w_{i,j}^{(l)}$ здійснюється за δ-правилом, де використовуються узагальнені помилки $\boldsymbol{\sigma}^{(l)}$. Очевидно, що для шару K помилка $\boldsymbol{\sigma}^{(K)} = \mathbf{e}_u(\mathbf{w}^{(k)})$. У методі ВР далі використовується локальний квадратичний функціонал $Q(\boldsymbol{\sigma}) = 0,5\ \boldsymbol{\sigma}^T\boldsymbol{\sigma}$. Мінімізація $Q(\boldsymbol{\sigma})$ по вагових коефіцієнтах $w_{i,j}^{(l)}$ $(l = K-1,...,1)$, що настроюються, методом градієнтного спуску з перерахунком помилки

$\boldsymbol{\sigma}^{(K)} = \mathbf{e}_u(\mathbf{w}^{(K)})$ для шару K в помилку $\boldsymbol{\sigma}^{(l)} = \mathbf{e}(\mathbf{w}^{(l)})$ для шарів $l = K-1,\ K-2,\ \ldots,1$ визначає сутність ВР-технології навчання багатошарових нейронних мереж.

В.3.1. Настройка коефіцієнтів вихідного шару.

Для шару K процедура настройки вектора всіх вагових коефіцієнтів в K-м шарі $\mathbf{w}^{(k)}$ відповідає зазвичай алгоритму градієнтного спуску:

$$\mathbf{w}^{(k)}(k+1) = \mathbf{w}^{(K)} - \gamma \nabla_w Q(\mathbf{e}_u(\mathbf{w}^{(k)}(k))) \qquad (В.1)$$

де:

$$\nabla_w Q(\cdot) = \mathbf{col}\left(\frac{\partial Q(\cdot)}{\partial \mathbf{w}_l^{(K)}},\ldots,\frac{\partial Q(\cdot 0)}{\partial \mathbf{w}_j^{(K)}},\ldots\frac{\partial Q(\cdot)}{\partial \mathbf{w}_{n_k}^{(K)}}\right)$$

вектор розмірністю $(n_k \times 1)$. Вектор $\mathbf{w}^{(k)}$ – складний:

$$\mathbf{w}^{(K)} = \mathrm{col}(\mathbf{w}_1^{(K)},\ldots,\mathbf{w}_i^{(K)},\ldots,\mathbf{w}_{n_K}^{(K)})$$

де:

$$\mathbf{w}^{(K)} = \mathrm{col}(w_{i,1}^{(K)},\ldots,w_{i,m}^{(K)},\ldots,w_{n_{K-1}}^{(K)})$$

$\mathbf{q}^{(K-1)}(k)$ – вектор розмірністю $((n_{K-1}+1)\times 1)$ базових елементів в шарі K:

$$\mathbf{q}^{(K-1)}(k) = \mathrm{col}(q_0^{(K-1)}, q_1^{(K-1)},\ldots,q_{n_{K-1}}^{(K-1)}).$$

Сигнал ініціалізації $q_0^{(K-1)}$ зазвичай приймається рівним +1,

Градієнт

$$\nabla_{\mathrm{w}} Q(\mathbf{e}_u) = \mathbf{e}_n^T \frac{\partial \mathbf{e}_u}{\partial \mathbf{w}} = -\mathbf{e}_u^T \frac{\partial \mathbf{q}^{(K)}}{\partial \mathbf{w}^{(K)}}$$

де похідна:

$$\frac{\partial \mathbf{q}^{(K)}}{\partial \mathbf{w}^{(K)}} = \frac{\partial \mathbf{f}(\mathbf{s}^{(K)})}{\partial \mathbf{w}^{(K)}} = \left(\frac{\partial \mathbf{s}^{(K)}}{\partial \mathbf{w}^{(K)}}\right)\left(\frac{\partial \mathbf{f}(\mathbf{s}^{(K)})}{\partial \mathbf{s}^{(K)}}\right)$$

є добуток матриць Якобі. перша з них, така, що розкривається в наступному вигляді:

$$\frac{\partial \mathbf{s}^{(K)}}{\partial \mathbf{w}^{(K)}} = \begin{bmatrix} \frac{\partial}{\partial \mathbf{w}_1^{(K)}} \\ \frac{\partial}{\partial \mathbf{w}_2^{(K)}} \\ \dots \\ \dots \\ \frac{\partial}{\partial \mathbf{w}_{n_K}^{(K)}} \end{bmatrix} \times [s_1^{(K)},\ s_2^{(K)},\ \dots, s_{n_K}^{(K)},] = \begin{bmatrix} \frac{\partial s_1^{(K)}}{\partial \mathbf{w}_1^{(K)}} & \dots & \frac{\partial s_{n_K}^{(K)}}{\partial \mathbf{w}_1^{(K)}} \\ \dots & \dots & \dots \\ \frac{\partial s_1^{(K)}}{\partial \mathbf{w}_i^{(K)}} & \dots & \frac{\partial s_{n_K}^{(K)}}{\partial \mathbf{w}_i^{(K)}} \\ \dots & \dots & \dots \\ \frac{\partial s_1^{(K)}}{\partial \mathbf{w}_{n_K}^{(K)}} & \dots & \frac{\partial s_{n_K}^{(K)}}{\partial \mathbf{w}_{n_K}^{(K)}} \end{bmatrix} = \operatorname{diag}\left(\frac{\partial s_i^{(K)}}{\partial \mathbf{w}_i^{(K)}} \right) -$$

квадратна матриця, кожен елемент якої на діагоналі є вектор-стовпець розмірністю $\left[(n_{K-1}+1)\times 1\right]|$, отже повна розмірність матриці рівна $(n_K(n_{K-1}+1)\times n_K)$ В матриці $\partial \mathbf{s}^{(K)}/\partial \mathbf{w}^{(K)}$ всі елементи, окрім діагональних, рівні нулю, а приватні похідні $\partial \mathbf{s}_i^{(K)}/\partial \mathbf{w}_i^{(K)}$ на діагоналі – всі однакові і рівно вектору $\mathbf{q}^{(K-1)}$. Це витікає з визначення дискримінантної функції $s_i^{(K)}$

$$s_i^{(K)} = \sum_{j=0}^{n_{K-1}} w_{i,j}^{(K)} q_j^{(K-1)} = \mathbf{w}_i^{(K)} \mathbf{q}^{(K-1)}.$$

Інша матриця Якобі розкривається у вигляді діагональної матриці $(n_K \times n_K)$

$$\frac{\partial \mathbf{f}(\mathbf{s}^{(K)})}{\partial \mathbf{s}^{(K)}} = \operatorname{diag}(f_s'(\mathbf{s}^{(K)})) = \begin{bmatrix} f_s'(\mathbf{s}_1^{(K)}) & 0 & 0 \\ \dots & \dots & \dots \\ 0 & 0 & f_s'(\mathbf{s}_{n_K}^{(K)}) \end{bmatrix}.$$

З урахуванням приведених матриць Якобі градієнт функціонала настройки K -го шару багатошарової нейронної мережі можна записати таким чином:

$$\nabla_w Q(\mathbf{e}_u) = -\left(\frac{\partial \mathbf{s}^{(K)}}{\partial \mathbf{w}^{(K)}} \right)\left(\frac{\partial \mathbf{f}(\mathbf{s}^{(K)})}{\partial \mathbf{s}^{(K)}} \right)\mathbf{e}_u = -\left(\frac{\partial \mathbf{s}^{(K)}}{\partial \mathbf{w}^{(K)}} \operatorname{diag}(f_s'(\mathbf{s}^{(K)})) \right)\mathbf{e}_u,$$

а алгоритм настройки вагових коефіцієнтів K-го шару має вигляд

$$\mathbf{w}^{(K)}(k+1) = \mathbf{w}^{(K)}(k) + \gamma\left(\frac{\partial \mathbf{s}^{(K)}}{\partial \mathbf{w}^{(K)}(k)}\operatorname{diag}(f_s'(\mathbf{s}^{(K)}(k))\right)\mathbf{e}_u(k+1)\ . \qquad \text{(B.2)}$$

Рекурентний алгоритм (В.2) можна розділити на n_K алгоритмів навчання i-x базових елементів, $i = \overline{1,\ n_K}$, оскільки діагональна матриця $\partial \mathbf{s}^{(K)} / \partial \mathbf{w}^{(K)}$ складається з однакових елементів $\mathbf{q}^{(K-1)}$.

$$\begin{aligned}\mathbf{w}_i^{(K)}(k+1) &= \mathbf{w}_i^{(K)}(k) + \gamma f_s'\ (s_i^{(K)}(k))e_{ui}(k+1)\mathbf{q}^{(k-1)}(k+1) = \\ &= \mathbf{w}_i^{(K)}(k) + \gamma\sigma_i(k+1)\mathbf{q}^{(K-1)}(k+1).\end{aligned} \qquad \text{(B.3)}$$

Узагальнена помилка $\boldsymbol{\sigma}$, навчання мережі для шару K рівна $f_s'\ (s_i^{(K)})e_{ui}$. У багатошарових мережах останній вихідний шар виконується зазвичай у вигляді мадалини з лінійними функціями активації і $f_s'\ (s) = 1$. Тоді $\sigma_i = e_{ui}$ і алгоритм настройки вагових коефіцієнтів шаруючи K приймає вигляд:

$$\mathbf{w}_i^{(K)}(k+1) = \mathbf{w}_i^{(K)}(k) + \gamma e_{ui}(k+1)\mathbf{q}^{(K-1)}(k+1)\,. \qquad \text{(B.4)}$$

У алгоритмах (В.2)-(В.4) $k = 1,\ 2,\ \ldots$ – номер інтеграції. Початкові значення вагових коефіцієнтів $w^{(K)}(0)$ задаються довільно, але бажано в околиці значень $w^{*(K)}$, для яких $Q(\mathbf{w}^*_{(K)}) = \min_{\mathrm{w}} Q(\mathbf{w}^{(K)})$.

В.3.2. Настройка вагових коефіцієнтів прихованих шарів нейронної мережі.

Для прихованих шарів $l = \overline{K-1,\ 1}$ помилки навчання в явному вигляді не можуть бути визначені. Проте очевидно, що якщо $\mathbf{e}_{\mathrm{u}} = \mathbf{u}^* - \mathbf{u} = \mathbf{u}^* - \mathbf{q}^{(K)} \neq 0$, то через архітектуру мережі існує ненульова узагальнена помилка і для всіх попередніх шарів. Суть методу ВР якраз і полягає в послідовному визначенні «при-

хованих» помилок, починаючи з обчислення помилки $\boldsymbol{\sigma}^{(K)}$ у вихідному шарі K, потім помилки $\boldsymbol{\sigma}^{(K-1)}$ в шарі $K-1$ і так до вхідного шару 1. Всі приховані помилки $\sigma^{(l)} = \boldsymbol{\sigma}(\mathbf{w}_i^l)$ залежать явно від вагових коефіцієнтів, що настроюються $\mathbf{w}_i^l$. Розглянемо суть необхідних перетворень в δ-правилі (2.2), яке використовує ідею обчислення прихованих помилок в рекурентній процедурі настройки вагових коефіцієнтів в шарах $l = \overline{K-1,1}$. З цією метою складемо алгоритм настройки вагових коефіцієнтів для окремих i-x базових елементів в шарі l:

$$\mathbf{w}_i^{(l)} = \mathbf{col}(w_{i,0}^{(l)},...,w_{i,j}^{(l)},...,w_{i,n_l}^{(l)}).$$

При виведенні алгоритму настройки вагових коефіцієнтів в шарах $K-1,\ K-2,\ ...\ ,1$ вкажемо, що в загальному випадку градієнт функціонала Q щодо вектора вагових коефіцієнтів $\mathbf{w}_i^{(l)}$ можна обчислити як:

$$\nabla \mathbf{w}_i^{(l)} Q = \frac{\partial \mathbf{s}^{(l)}}{\partial \mathbf{w}_i^{(l)}} \frac{\partial \mathbf{q}^{(l)}}{\partial \mathbf{s}^{(l)}} \frac{\partial Q}{\partial \mathbf{q}^{(l)}}$$

Тут, як і для шару K, вважаємо для спільності і для спрощення подальших перетворень вектор $\mathbf{q}^{(l)} = \mathrm{col}(q_0^{(l)},\ q_1^{(l)},\ ...\ ,q_{n_l}^{(l)})$.

У практичних застосуваннях $q_0^{(l)} = +1$. У подальшому виводі вважатимемо, що ініціалізуючі сигнали $q_0^{(l)}$ є виходами фіктивних, «нульових» базових елементів з функціями активації $f(s_0^{(l)})$ того ж типу, що і в реальних елементах. Тоді випадку $q_0^{(l)} = +1$ відповідає робоча точка $f(s_0^{(l)}) = 1$.

З урахуванням зробленого припущення розмірність матриці Якобі $\partial \mathbf{s}^{(l)} / \partial \mathbf{w}_i^{(l)}$, де $\mathbf{s}^{(l)} = \mathrm{col}(s_0^{(l)}, s_1^{(l)}, ..., s_{n_l}^{(l)})$, рівна $(n_{l-1}+1)\times(n_l+1)$, відповідно розмірність матриці Якобі $\partial \mathbf{q}^{(l)} / \partial \mathbf{s}^{(l)}$ – $\dim \mathrm{diag}\,(f_s'(s_0^{(l)})) = (n_l+1)\times(n_l+1)$. Якщо покласти $f_s'(s_0^{(l)}) = 1$, то верхній діагональний елемент матриці $\partial \mathbf{q}^{(l)} / \partial \mathbf{s}^{(l)}$ буде рівний нулю. Отже, матриці

$$\left(\frac{\partial \mathbf{s}^{(l)}}{\partial \mathbf{w}_i^{(l)}}\right) = \left\| \frac{\partial s_i^{(l)}}{\partial w_{i,j}^{(l)}} \right\|, \quad i = \overline{0, n_l}, \quad j = \overline{0, n_{l-1}};$$

$$\left(\frac{\partial \mathbf{q}^{(l)}}{\partial \mathbf{s}^{(l)}}\right) = \mathrm{diag}(f_s'(s^{(l)}))$$

складені. Позначимо першу з них символом $\mathbf{S}^{(l)}$, другу $\mathbf{F}^{(l)}$. Елементами матриці служать приватні похідні $\partial s_i^{(l)} / \partial w_{i,j}^{(l)}$. Оскільки

$$s_i^{(l)} = \sum_{j=0}^{n_{l-1}} w_{i,j}^{(l)} q_j^{(l-1)},$$

матриця $\mathbf{S}^{(l)}$ має наступну структуру:

$$\mathbf{S}^{(l)} = \begin{bmatrix} q_0^{(l-1)} & q_0^{(l-1)} & \dots & q_0^{(l-1)} \\ q_1^{(l-1)} & q_1^{(l-1)} & \dots & q_1^{(l-1)} \\ \dots & \dots & \dots & \dots \\ q_{n_{j-1}}^{(l-1)} & q_{n_{j-1}}^{(l-1)} & \dots & q_{n_{j-1}}^{(l-1)} \end{bmatrix} = \left[\mathbf{q}^{(l-1)} \vdots \mathbf{q}^{(l-1)} \vdots \cdots \vdots \mathbf{q}^{(l-1)} \right],$$

розмірність $\mathbf{S}^{(l)}$ рівна $(n_{l-1}+1)\times(n_{l-1}+1)$ і $\mathbf{q}^{(l-1)} = \mathrm{col}(q_0^{(l-1)},...,q_{n_{l-1}}^{(l-1)})$.

Елементами діагональної $\mathbf{F}^{(l)}$ матриці є приватні похідні функцій активації $f(s)$ по своєму аргументу $s_i (i = \overline{0,\ n_l})$. Можна показати, що добуток матриць $\mathbf{S}^{(l)}$ і $\mathbf{F}^{(l)}$ зводиться до зовнішнього добутку векторів $\mathrm{q}^{(l-1)}$ і $\mathbf{f}_s'(s^{(l)}) = (f_s'(s_0^{(l)},...,f_s'(s_{n_l}^{(l)}))$.

$$\mathbf{S}^{(l)}\mathbf{F}^{(l)} = \mathbf{q}^{(l-1)}(\mathbf{f}_s'(\mathbf{s}^{(l)}))^T.$$

Похідна скалярного функціонала $Q(\boldsymbol{\sigma})$ по векторному аргументу $\mathbf{q}^{(l)} (l = \overline{K-1,\ 1}$ по своєму сенсу є вектор чутливості функціонала Q до зміни вектора виходу $\mathbf{q}^{(l)}$ l-го шару мережі. Для шару l введемо вектор чутливості функціонала Q:

$$\Lambda^{(l)} = \frac{\partial Q}{\partial \mathbf{q}^{(l)}} = \mathrm{col}\left(\frac{\partial Q}{\partial q_0^{(l)}}, \frac{\partial Q}{\partial q_1^{(l)}}, ..., \frac{\partial Q}{\partial q_{n_l}^{(l)}} \right) = \mathbf{col}(\lambda_0^{(l)}, \lambda_1^{(l)}, ..., \lambda_{n_l}^{(l)}) .$$

Приватні функції чутливості $\lambda_i^{(l)}$ обчислюються, по-перше, відповідно до правила диференціювання складних функцій; по-друге, відповідно до структури багатошарової нейронної мережі і мають вигляд:

$$\lambda_i^{(l)} = \frac{\partial Q}{\partial q_i^{(l)}} = \sum_{j=1}^{n_{l+1}} \frac{\partial Q}{\partial q_j^{(l+1)}} \frac{\partial q^{(l+1)}}{\partial q_i^{(l)}} = \sum_{j=1}^{n_{l+1}} \lambda_j^{(l+1)} d_{j,i}^{(l+1)} = (\Lambda^{(l+1)})^T \mathbf{d}_i^{(l+1)},$$

$$i = \overline{1,\ n_l}.$$

Повний вектор функцій чутливості функціонала Q до зміни вектора виходу нейронів l-го шару є

$$\Lambda^{(l)} = \frac{\partial Q}{\partial \mathbf{q}^{(l)}} = \mathbf{D}^{(l+1)} \Lambda^{(l+1)},\ \ l = \overline{K-1,\ 1}. \qquad \text{(B.5)}$$

Матрицю $\mathbf{D}^{(l+1)}$ назвемо матрицею переходу у зворотному напрямі, оскільки для розрахунку функції $\Lambda^{(l)}$ використовуються функції чутливості $\Lambda^{(l+1)}$ і $\mathbf{D}^{(l+1)}$ подальшого шару. Матриця $\mathbf{D}^{(l+1)}$ складена з вектор-рядків $(\mathbf{d}_i^{(l+1)})^T$ де $i = \overline{0,\ n_l}$. Нагадаємо, що за умовчанням $q_0^{(l)} = +1$, а вагові коефіцієнти ініціалізуючих входів $w_{j,0}^{(l+1)}$ всіх базових елементів в шарі $l+1$ також настроюються. Елементами векторів за визначенням є приватні похідні:

$$\frac{\partial q_j^{(l+1)}}{\partial q_i^{(l)}} = \frac{\partial}{\partial q_i^{(l)}} f\left(\sum_{j=0}^{nl} w_{j,i}^{(l+1)} q_i^{(l)} \right) = \left(\frac{\partial f(s_j^{(l+1)})}{\partial s_j^{(l+1)}} \right) \left(\frac{\partial s_j^{(l+1)}}{\partial q_j^{(l)}} \right) = f_s'(s_j^{(l+1)}) w_{j,i}^{(l+1)} .$$

Для всієї множини індексів $i = \overline{0.\ n_l},\ j = \overline{1,\ n_{l+1}}$ матриця переходу $\mathbf{D}^{(l+1)}$ має наступну структуру:

$$\mathbf{D}^{(l+1)} = \frac{\partial \mathbb{q}^{(l+1)}}{\partial \mathbf{q}^{(l)}} = \begin{bmatrix} \dfrac{\partial}{\partial q_0^{(l)}} \\ \dfrac{\partial}{\partial q_1^{(l)}} \\ \cdots \\ \dfrac{\partial}{\partial q_{n_l}^{(l)}} \end{bmatrix} \times (q_0^{(l+1)}, q_1^{(l+1)}, ..., q_{n_{l+1}}^{(l+1)}) = \begin{bmatrix} \dfrac{\partial q_0^{(l+1)}}{\partial q_0^{(l)}} & \dfrac{\partial q_1^{(l+1)}}{\partial q_0^{(l)}} & \cdots & \dfrac{\partial q_{n_{l+1}}^{(l+1)}}{\partial q_0^{(l)}} \\ \dfrac{\partial q_0^{(l+1)}}{\partial q_1^{(l)}} & \dfrac{\partial q_1^{(l+1)}}{\partial q_1^{(l)}} & \cdots & \dfrac{\partial q_{n_{l+1}}^{(l+1)}}{\partial q_1^{(l)}} \\ \cdots & \cdots & \cdots & \cdots \\ \dfrac{\partial q_0^{(l+1)}}{\partial q_{n_l}^{(l)}} & \dfrac{\partial q_1^{(l+1)}}{\partial q_{n_l}^{(l)}} & \cdots & \dfrac{\partial q_{n_{l+1}}^{(l+1)}}{\partial q_{n_l}^{(l)}} \end{bmatrix}.$$

Відзначимо, що розмірність матриці переходу $\mathbf{D}^{(l+1)}$ рівна $\left[(n_{l+1}+1)\times(n_1+1)\right]$. Приватні похідні в матриці $\mathbf{D}^{(l+1)}$ визначені раніше. Тепер можна скласти вектор-рядки $(\mathbf{d}_i^{(l+1)})^T$:

$$(\mathbf{d}_i^{(l+1)})^T = (f_s'(s_0^{(l+1)})w_{0,i}^{(l+1)}, f_s'(s_1^{(l+1)})w_{1,i}^{(l+1)}, ..., f_s'(s_{n_{l+1}}^{(l+1)})w_{n_{l+1},i}^{(l+1)})$$

розмірністю $\left[1\times(n_{l+1}+1)\right]$. Введемо матриці:

а)

$$\mathbf{W}^{(l+1)} = \left[\mathbf{w}_0^{(l+1)} \vdots \mathbf{w}_1^{(l+1)} \vdots \cdots \vdots \mathbf{w}_{n_{l+1}}^{(l+1)}\right] = \left\|w_{i,j}^{(l)}\right\|, \;\; i = \overline{0,\; n_{l+1}}; \;\; j = \overline{0,\; n_l},$$

тобто

$$\dim \mathbf{W}^{(l+1)} = (n_l+1)\times(n_{l+1}+1),$$

де

$$\mathbf{w}_i^{(l+1)} = \mathrm{col}(w_{i,0}^{(l+1)}, ..., w_{i,j}^{(l+1)}, ..., w_{i,n_l}^{(l+1)});$$

б)

$$\mathbf{F}^{(l+1)} = \mathrm{diag}(f_s'(\mathbf{s}^{(l+1)})) = \begin{bmatrix} f_s'(s_0^{(l+1)}) & ... & 0 & 0 \\ 0 & f_s'(s_1^{(l+1)}) & ... & ... \\ ... & ... & ... & ... \\ 0 & 0 & ... & f_s'(s_{n_{j-1}}^{(l+1)}) \end{bmatrix}$$

$$\dim \mathbf{F}^{(l+1)} = (n_{l+1}+1)\times(n_{l+1}+1).$$

Відмітимо, що для фіктивного «нульового» базового елементу $q_0^{(l+1)} = +1$ і елемент матриці $\mathbf{F}^{(l+1)} f_s'(s_0^{(l+1)}) = 0$. З урахуванням введених матриць $\mathbf{W}^{(l+1)}$ і $\mathbf{F}^{(l+1)}$ матрицю $\mathbf{D}^{(l+1)}$ запишемо у вигляді:

$$\mathbf{D}^{(l+1)} = \left\| \frac{\partial q_j^{(l+1)}}{\partial q_i^{(l)}} \right\|_{i=\overline{0,\ n_l}}^{j=\overline{0,\ n_{l+1}}} = (\mathbf{W}^{(l+1)})^T \mathbf{F}^{(l+1)} = (\mathbf{W}^{(l+1)})^T \text{ diag } (f_s'(s^{(l+1)})). \qquad \text{(B.6)}$$

Тепер можна записати формулу обчислення градієнта функціонала навчання багатошарової нейронної мережі $Q(\boldsymbol{\sigma}) = 0{,}5\boldsymbol{\sigma}^T\boldsymbol{\sigma}$ по вагових коефіцієнтах будь-якого шару $l = \overline{K-1,\ 1}$:

$$\nabla \mathbf{w}_i^{(l)} Q = \mathbf{S}^{(l)} \mathbf{F}^{(l)} \Delta^{(l)} = \mathbf{q}^{(l-1)} (\mathbf{f}_s'(\mathbf{s}^{(l)}))^T \Lambda^{(l)} \qquad \text{(B.7)}$$

Повний алгоритм навчання багатошарової нейронної мережі по методу ВР включає початковий алгоритм настройки вагових коефіцієнтів K-го шару (В.4) і алгоритми настройки базових елементів шарів $l = \overline{K-1,\ 1}$ використовують формулу (В.7).

В.3.3. Алгоритм навчання багатошарової нейронної мережі по методу зворотного розповсюдження помилки.

Приведемо алгоритм навчання нейронної мережі методом зворотного розповсюдження помилки в одній з можливих форм, витікаючої з прийнятої методики виводу алгоритму:

а)

$$\mathbf{w}_i^{(K)}(k+1) = \mathbf{w}_i^{(K)}(k) + \gamma f_s'(s_i^{(K)}(k+1)) e_{ui}(k+1) \mathbf{q}^{(K-1)}(k+1) =$$

$$\mathbf{w}_i^{(K)}(k) + \gamma \sigma_i^{(K)}(k+1) \mathbf{q}^{(K-1)}(k+1);$$

$$\gamma > 0; \quad k = 0,1,2...; \quad e_{ui} = u_i^* - u_i; \quad i \in \overline{1, n_K}; \qquad \text{(B.8)}$$

$$q^{(K-1)} = \text{col}(q_0^{(K-1)}, q_1^{(K-1)}, ..., q_j^{(K-1)}, ..., q_{n_{K-1}}^{(K-1)});$$

б)

$$\mathbf{w}_i^{(l)}(k+1) = \mathbf{w}_i^{(l)}(k) - \gamma \mathbf{q}^{(l-1)}(k+1)(\mathbf{f}_s'(s_i^{(l)}(k+1)))^T \Lambda^{(l)}(k);$$

$$\Lambda^{(l)}(k) = \mathbf{D}^{(l+1)}\Lambda^{(l+1)}(k);$$

$$\mathbf{D}^{(l+1)}(k) = (\mathbf{W}^{(l+1)}(k))^T \operatorname{diag}(f_s'(\mathbf{s}^{(l)}(k)));$$

$$\gamma > 0;\ \ l = \overline{K-1,\ 1};\ \mathbf{q}^{(l-1)} = \operatorname{col}(q_0^{(l-!)}, q_1^{(l-!)}, ..., q_{l_{K-1}}^{(l-!)});\ i = \overline{0,\ n_l}$$

(B.9)

Зокрема, алгоритм настройки нейронів шару $K-1$ у відповідності до пункту б) має вид

$$\mathbf{w}_i^{(K-1)}(K+1) = \mathbf{w}_i^{(K-1)}(k) - \gamma \mathbf{q}^{(K-2)}(k+1)(\mathbf{f}_s'(s_i^{(K-1)}(k+1))))^T \Lambda^{(K-1)}(k),$$

$$\Lambda^{(K-1)}(k) = \mathbf{D}^{(K)}(k)\Lambda^{(K)}(k);$$

$$\mathbf{D}^{(K)}(k) = \mathbf{W}^{(K)}(k)\operatorname{diag}(f_s'\ (\mathbf{s}^{(K)}(k))) = \mathbf{W}^{(K)}(k);$$

$$\Lambda^{(K)}(k) = \operatorname{col}\left(\frac{\partial Q}{\partial \mathbf{q}_1^{(K)}}, ..., \frac{\partial Q}{\partial \mathbf{q}_{n_K}^{(K)}}\right) = -\mathbf{e}_u(k).$$

тоді

$$\Lambda^{(K-1)}(k) = -\mathbf{W}^{(K)}(k)\mathbf{e}_u(k),$$

$$(\mathbf{f}_s'(s_i^{(K-1)}(k+1))))^T \Lambda^{(K-1)}(k) =$$

$$= -(\mathbf{f}_s'(s_i^{(K-1)}(k+1)))^T \mathbf{W}^{(K)}(k)\mathbf{e}_u^T(k) = \sigma_1^{(K-1)}(k+1)$$

– скалярна узагальнена помилка навчання нейромережі для i-го базового елементу в шарі $K-1$. З її використанням отримуємо алгоритм настройки, формою в точності відповідний δ-правилу:

$$\mathbf{w}_i^{(K-1)}(k+1) = \mathbf{w}_i^{(K-1)}(k) - \gamma\sigma_i^{(K-1)}(k)\mathbf{q}^{(K-2)}(k+1);$$

$$\sigma_i^{(K-1)}(k+1) = -(\mathbf{f}_s'(s_i^{(K-1)}(k)))^T \mathbf{W}^{(K)}(k)\mathbf{e}^{(K)}(k+1); \qquad \text{(B.10)}$$

$$\mathbf{e}_u^{(K)}(k) = \mathbf{u}^*(k) - \mathbf{q}^{(K)}(k).$$

Аргумент k позначає момент використання відповідних значень заданих множин $\mathbf{u}^*$ і $\mathbf{r} = \mathbf{q}^{(0)} = \mathbf{r}$ на k-й ітерації обчислень вагових коефіцієнтів $\mathbf{w}_i^{(l)}$ на «уроках» навчання, число яких залежить від числа шарів мережі і кількості нейронів в шарах.

Фаза навчання завершена, коли досягнутий для заданих в своєму класі множин $\mathbf{u}^*$ і $\mathbf{r} = \mathbf{q}^{(0)} = \mathbf{r}$. Мережа фіксує цей стан в значеннях своїх вагових коефіцієнтів і відповідних «віртуальних» зсувах центрів симетрії функцій активації і їх похідних активованих базових елементів і зберігає цей стан після зняття сигналу $\mathbf{q}^{(0)} = \mathbf{r}$ на входах мережі а виді віртуальної статичної нелінійної характеристики, зібраної з фрагментів сигмоїдних функцій. Якщо тепер па входах пред'явити ту ж множину або близьку до неї з того ж класу, то нейронна мережа практично миттєво відновить на виході ту ж множину $\mathbf{u} = \mathbf{q}^{(K)}$, що і після фази навчанні. Цей етап функціонування мережі називають фазою спогаду. Мережа, таким чином, виконує функцію асоціативної пам'яті. Якщо ж множини u^*, $\mathbf{q}^{(0)} = \mathbf{r}$ є функціями часу: $\mathbf{q}^{(0)} = \mathbf{r}(t)$, $\mathbf{u}^* = \mathbf{u}^*(t)$, то стандартний алгоритм ВР стає неавтономним і вимагає завдання всієї множини значень змінних $\mathbf{r}(t)$ і $\mathbf{u}^*(t)$. Неминуче зростання числа шарів і базових елементів в них і пов'язане з цим збільшення тривалості навчання викликає необхідність забезпечення стійкості процесу настройки мережі по алгоритму (В.8) (В.9).

Очевидно, що в завданнях управління $\mathbf{u}^*(t)$ – функція оптимального управління динамічним об'єктом. Управління за допомогою багатошарової нейронної мережі в двохетапному режимі можливо, якщо тільки функція оптимального управління наперед розрахована і може бути визначена вся множина

значень помилки навчання мережі $\mathbf{e}_u(t)$. на інтервалі її визначення. Використання іншої вимірювальної інформації, наприклад помилки відтворення заданої функції $\mathbf{r}(t)$ на виході динамічного об'єкту, вносить запізнювання до алгоритму настройки і приводить до нестійкого процесу. Виникає проблема синтезу динамічних алгоритмів навчання багатошарової нейронної мережі, що враховують вплив інерційності, що вноситься до процесу настройки по алгоритму ВР.

В.4 Перенавчання і узагальнення нейронних мереж

Одна з найбільш серйозних труднощів викладеного підходу полягає в тому, що таким чином ми мінімізуємо не ту помилку, яку насправді потрібно мінімізувати, – помилку, яку можна чекати від мережі, коли їй подаватимуться абсолютно нові спостереження. Інакше кажучи, ми хотіли б, щоб нейронна мережа володіла здатністю узагальнювати результат на нові спостереження. Насправді мережа навчається мінімізувати помилку на навчальній множині, і у відсутність ідеальної і нескінченно великої навчальної множини це зовсім не те ж саме, що мінімізувати «справжню» помилку на поверхні помилок в наперед невідомій моделі явища.

Найсильніше ця відмінність виявляється в проблемі перенавчання. При дуже близькій підгонці явище простіше буде продемонструвати не для нейронної мережі, а на прикладі апроксимації за допомогою поліномів, при цьому суть явища абсолютно та ж.

Поліном (або многочлен) – це вираз, що містить тільки константи і цілі ступені незалежної змінної. Графіки поліномів можуть мати різну форму. Чим вище ступінь многочлена (і, тим самим, ніж більше членів в нього входить), тим більше складною може бути ця форма. Якщо у нас є деякі дані, ми можемо поставити мету підігнати до них поліноміальну криву (модель) і отримати, таким чином, пояснення для наявної залежності. Проте наші дані можуть бути зашумлені, тому не можна вважати, що сама краща модель задається кривою, яка в точності проходить через все наявні крапки. Поліном низького порядку

може бути недостатньо гнучким засобом для апроксимації даних, тоді як поліном високого порядку може виявитися занадто гнучким і буде точно слід даним, приймаючи при цьому хитромудру форму, що не має ніякого відношення до форми справжньої залежності.

Нейронна мережа стикається з точно такою ж трудністю. Мережі з великим числом вагів моделюють складніші функції і, отже, схильні до перенавчання. Мережа ж з невеликим числом вагів може виявитись недостатньо гнучкою, щоб змоделювати наявну залежність. Наприклад, мережа без проміжних шарів насправді моделює звичайну лінійну функцію.

Як же вибрати «правильний» ступінь складності для мережі? Складніша мережа майже завжди дає меншу помилку, але це може свідчити не про хорошу якість моделі, а про перенавчання.

Відповідь полягає в тому, щоб використовувати механізм контрольної перехресної перевірки (крос-перевірки), при якому частина навчальних спостережень резервується і не використовується в процесі навчання по алгоритму зворотного розповсюдження. Натомість у міру роботи алгоритму вона використовуються для незалежної перевірки результату. На самому початку роботи помилка мережі на навчальній і перевірочній множинах буде однаковою, але у міру того як мережа навчається, помилка навчання, природно, убуває, і, поки навчання зменшує дійсну функцію помилок, помилка на контрольній (перевірочній) множині також убуватиме. Якщо ж контрольна помилка перестала убувати або навіть стала рости, це указує на те, що мережа почала дуже близько апроксимувати дані і навчання слід зупинити.

Відмічене явище занадто точної апроксимації в процесі навчання і називається перенавчанням. Якщо таке трапляється, то рекомендується зменшити число прихованих елементів і/або шарів, бо мережа є дуже могутньою для даного завдання. Якщо ж мережа, навпаки, була узята недостатньо складною для того, щоб моделювати наявну залежність, то перенавчання, швидше за все, не відбудеться, і обидві помилки – навчання і перевірки – не досягнуть достатнього рівня трохи.

Описані проблеми з локальними мінімумами і вибором розміру мережі призводять до того, що при практичній роботі з нейронними мережами, як правило, доводиться експериментувати з великим числом різних мереж, деколи навчаючи кожну з них по кілька разів (щоб не бути введеним в оману локальними мінімумами) і порівнюючи отримані результати. Головним показником якості результату тут є контрольна помилка. При цьому, відповідно до загальнонаукового принципу, згідно якому при інших рівних слід віддати перевагу простішій моделі, з двох мереж з приблизно рівними помилками контролю має сенс вибрати ту, яка менше.

Необхідність багатократних експериментів веде до того, що перевірочна множина починає грати ключову роль у виборі моделі, тобто стає частиною процесу навчання. Тим самим ослабляється її роль як незалежного критерію якості моделі – при великому числі експериментів є ризик вибрати «вдалу» мережу, що дає добрий результат на перевірочній (контрольному) множині.

Для того, щоб додати остаточній моделі належну надійність, часто (принаймні, коли об'єм навчальних даних це дозволяє) поступають так: резервують ще одну – тестову множину спостережень. Підсумкова модель тестується на даних з цієї множини, щоб переконатися, що результати, досягнуті на навчальній і контрольній множинах, реальні. Зрозуміло, для того, щоб добре грати свою роль, тестова множина повинна бути використана тільки один раз: якщо її використовувати повторно для коректування процесу навчання, то вона фактично перетвориться на контрольну множину.

ДОДАТОК Г
УЗАГАЛЬНЕНЕ НЕЙРОННЕ УПРАВЛІННЯ З ПРОГНОЗОМ

Г.1 Алгоритм нейроуправління з прогнозом

У роботі використовується ефективна реалізація узагальненого управління з прогнозом (Generalized Predictive Control, GPC) з використанням багатошарової прямонаправленої нейронної мережі, як нелінійній моделі об'єкту управління [70]. Завдяки використанню оптимізаційного алгоритму Ньютона-Рафсона, число ітерацій, необхідних для збіжності, значно менше, ніж при використанні інших методів. Головні витрати алгоритму Ньютона-Рафсона в обчисленні Гесиана, але навіть з цими додатковими розрахунками низьке число ітерацій робить алгоритм Ньютона-Рафсона швидшим, ніж інші методи, при цьому він може використовуватися для управління в режимі реального часу (real-time control).

Розглянемо стисло основні положення узагальненого алгоритму нейроуправления з прогнозом (Neural Generalized Predictive Control algorithm) із застосуванням оптимізаційного алгоритму Ньютона-Рафсона.

Узагальнене управління з прогнозом, введене Кларком і його співробітниками в 1987, належить до класу методів цифрового управління, званих модель-орієнтоване управління з прогнозом (Model-Based Predictive Control, MBPC) [71-77]. MBPC техніки були успішно проаналізовані і упроваджені в процес управління виробництвом в кінці 1970-х і продовжують використовуватися, оскільки вони системно враховують обмеження реальних об'єктів в режимі реального часу. GPC застосовуються в управлінні немінімально фазовими об'єктами, нестійкими об'єктами і об'єктами із змінним або невідомим часом затримки. Узагальнене управління з прогнозом також є робасним по відношенню до помилок моделювання, перевизначення і недовизначення значень параметрів і шумів датчиків [71]. GPC було спочатку розроблене для моделей лінійних об'єктів, що приводить до рівнянь, які можуть бути вирішені аналіти-

чно. При використанні нелінійних моделей необхідні нелінійні оптимізаційні алгоритми. Це впливає на якість і ефективність обчислень, що визначають вхідні сигнали, що управляють. Для нелінійних об'єктів здатність GPC робити точні прогнози може бути вдосконалена, якщо для вивчення динаміки об'єкту замість стандартних методів нелінійного моделювання застосовувати нейронні мережі. Вибір алгоритму мінімізації впливає на обчислювальну ефективність алгоритму. Як було вказано, при використанні для оптимізації алгоритму Ньютона-Рафсона число ітерацій, необхідних для збіжності, істотно менше, ніж при використанні інших методів.

Схема системи узагальненого нейроуправління з прогнозом (Neural Generalized Predictive Control, NGPC) показана на рис. Г.1.

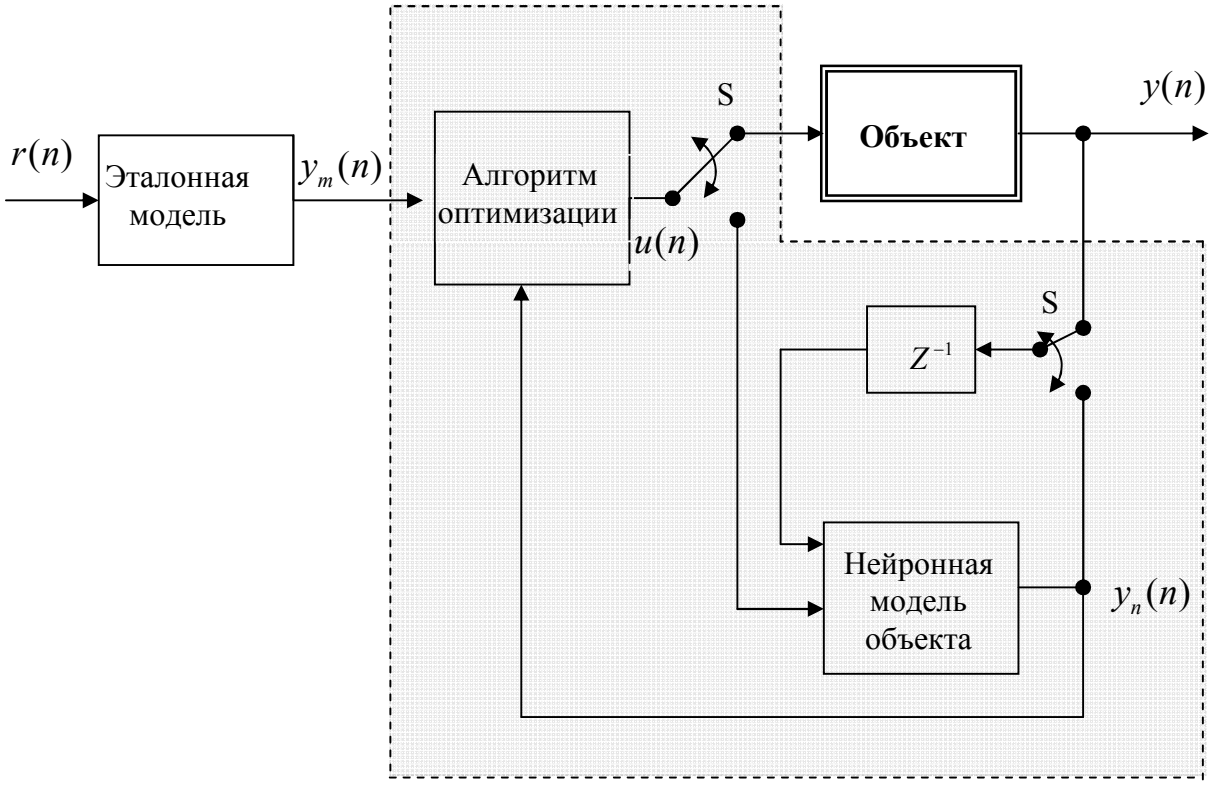


Рисунок Г.1 – Схема системи узагальненого нейроуправления з прогнозом

Вона складається з чотирьох компонент: керованого об'єкту, еталонної моделі, яка описує бажану якість об'єкту, що моделює об'єкт нейронної мережі і алгоритму мінімізації функціонала якості (Cost Function Minimization, CFM), що

визначає вхідний сигнал, необхідний для досягнення бажаної поведінки об'єкту. NGPC алгоритм складається з блоку CFM і блоку нейронної мережі.

Принцип роботи системи полягає в наступному. Вхідний сигнал $r(n)$ подається еталонній моделі. Ця модель видає еталонний сигнал $y_m(n)$, який служить входом для CFM блоку. CFM алгоритм розраховує сигнал, який служить входом для об'єкту або моделі об'єкту. Двополюсною двохпозиційний перемикач S встановлюється в положення до об'єкту, коли CFM алгоритм використовується для визначення оптимального вхідного сигналу $u(n)$, який мінімізує вибраний критерій якості управління. Між тактами перемикач встановлений в положення до моделі об'єкту, де CFM алгоритм використовує цю модель для розрахунку наступного входу, що управляє $u(n+1)$ шляхом прогнозу у відповідь сигналу, отриманого від моделі об'єкту. Як тільки функціонал якості мінімізований, цей вхідний сигнал подається на об'єкт.

Узагальнений алгоритм нейроуправління з прогнозом складається з наступних основних кроків.

1) Генерується задаюча траєкторія. Якщо майбутня траєкторія $y_m(n)$ невідома, вважається $y_m(n) = const$.

2) Використовуючи заздалегідь розрахований управліючий вхідний вектор і нейронну модель об'єкту управління, виконується прогноз поведінки об'єкту.

3) Розраховується новий управляючий вхідний сигнал, що мінімізує функціонал якості.

4) Повторюються кроки 2 і 3, поки не буде досягнута необхідна мінімізація.

5) Посилається перший управляючий вхідний сигнал на об'єкт.

6) Повторюється весь процес для кожного тимчасового кроку.

Якість обчислення при реалізації узагальненого управління з прогнозом в основному залежить від вибору мінімізуючого алгоритму для CFM блоку. Існують декілька алгоритмів мінімізацій, які реалізуються в CFM, такі як: Nongradient, Simplex і Successive Quadratic Programming. Вибір методу мінімізації

може бути заснований на наступних критеріях: число ітерацій, обчислювальні витрати і точність рішення. У загальному випадку, ці алгоритми вимагають великого числа ітерацій, що робить управління в режимі реального часу скрутним. Дуже невелике число публікацій присвячені реалізаціям управління в режимі реального часу для об'єктів, що мають великі постійні часу. Для можливості використання в системах з об'єктами, що мають малі постійні часу, потрібний швидший оптимізаційний алгоритм. Алгоритм Ньютона-Рафсона володіє квадратичною збіжністю, тоді як інші алгоритми мають нижчу швидкість збіжності. Вища швидкість збіжності алгоритму Ньютона-Рафсона вимагає високих обчислювальних витрат, але виправдовується показником швидкості збіжності.

Якість обчислення при реалізації узагальненого управління з прогнозом в основному залежить від вибору мінімізуючого алгоритму для CFM блоку. Існують декілька алгоритмів мінімізацій, які реалізуються в CFM, такі як: Non-gradient [74], Simplex [76] і Successive Quadratic Programming [76,77]. Вибір методу мінімізації може бути заснований на наступних критеріях: число ітерацій, обчислювальні витрати і точність рішення. У загальному випадку, ці алгоритми вимагають великого числа ітерацій, що робить управління в режимі реального часу скрутним. Дуже невелике число публікацій присвячені реалізаціям управління в режимі реального часу для об'єктів, що мають великі постійні часу [76,77]. Для можливості використання в системах з об'єктами, що мають малі постійні часу, потрібний швидший оптимізаційний алгоритм. Алгоритм Ньютона-Рафсона володіє квадратичною збіжністю, тоді як інші алгоритми мають нижчу швидкість збіжності. Вища швидкість збіжності алгоритму Ньютона-Рафсона вимагає високих обчислювальних витрат, але виправдовується показником швидкості збіжності.

Як указувалося вище, узагальнений алгоритм нейроуправління з прогнозом ґрунтується на мінімізації функціонала якості на кінцевому діапазоні прогнозів. Функціонал якості, використовуваний в дисертаційній роботі, має наступний вигляд:

$$J = \sum_{j=N_1}^{N_2} [y_m(n+j) - y_n(n+j)]^2 + \sum_{j=1}^{N_u} \lambda(j)[\Delta u(n+j)]^2 \ , \qquad (\Gamma.1)$$

де N_1 – нижня межа прогнозу;

N_2 – верхня межа прогнозу;

N_u – діапазон управління;

y_m – бажана траєкторія;

y_n – передбачений вихід нейронної мережі;

λ – ваговий множник;

$\Delta u(n+j)$ – зміна u визначуване таким чином:

$$u(n+j) - u(n+j-1).$$

Цей функціонал мінімізує не тільки середньоквадратичну помилку між еталонним сигналом і сигналом, що видається моделлю об'єкту, але також зважену середньоквадратичну швидкість зміни сигналу, що управляє.

Коли функціонал якості мінімізований, генерується вхідний управляючий сигнал, що дозволяє об'єкту відстежувати задану траєкторію з деякою точністю. Функціонал якості містить чотири параметри, що настроюються N_1, N_2, N_u, λ. Параметри N_1 і N_2 задають межі, усередині яких обчислюється помилка стеження. N_u є межею діапазону управління. Єдине обмеження, що накладається на значення N_u і N_1 наступне: вони повинні бути менше або рівні N_2. Друга сума містить ваговий множник λ, який введений для управління балансом між першою і другою сумою.

Для реалізації оптимального управління, функціонал якості мінімізується за допомогою алгоритму Ньютона-Рафсона. Мета алгоритму – мінімізувати функціонал J (Г.1) по вектору $[u(n+1),\ u(n+2),\ u(n+N_u)]^T$. Позначимо даний вектор як $\mathbf{U}$. Використовуючи як алгоритм CFM алгоритм Ньютона-Рафсона, ітераційно мінімізується функціонал J для визначення оптимального $\mathbf{U}$. Ітераційний процес визначає значення J на кожній ітерації, яке позначимо як $J(k)$. Для

кожної ітерації розраховується також значення управляючого вектора

$$\mathbf{U}(k)=\begin{bmatrix} u(n+1) \\ u(n+2) \\ \vdots \\ u(n+N_u) \end{bmatrix},$$

де k=1, ... – номер ітерації.

Правило оновлення $\mathbf{U}(k+1)$ у алгоритмі Ньютона-Рафсона виглядає таким чином:

$$\mathbf{U}(k+1)=\mathbf{U}(k)-\left(\frac{\partial^2\mathbf{J}}{\partial\mathbf{U}^2}(k)\right)^{-1`}\frac{\partial\mathbf{J}}{\partial\mathbf{U}}(k) \qquad (\Gamma.2)$$

де якобіан позначений, як

$$\frac{\partial\mathbf{J}}{\partial\mathbf{U}}(k)=\begin{bmatrix} \dfrac{\partial J}{\partial u(n+1)} \\ \vdots \\ \dfrac{\partial J}{\partial u(n+N_u)} \end{bmatrix},$$

а гесиан, як

$$\frac{\partial^2\mathbf{J}}{\partial\mathbf{U}^2}(k)=\begin{bmatrix} \dfrac{\partial^2 J}{\partial u(n+1)^2} & \cdots & \dfrac{\partial^2 J}{\partial u(n+1)\partial u(n+N_u)} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial^2 J}{\partial u(n+N_u)\partial u(n+1)} & \cdots & \dfrac{\partial^2 J}{\partial u(n+N_u)^2} \end{bmatrix}.$$

Рішення рівняння (8.2) вимагає явного звернення матриці Гессе. Цей процес може зажадати великих обчислювальних витрат. Одним з підходів, використовуваних, щоб уникнути звернення матриці, служить LU декомпозиція [78] для отримання вхідного вектора $\mathbf{U}(k+1)$. Це досягається шляхом запису рівняння (Г.2) у вигляді системи лінійних рівнянь $\mathbf{Ax=b}$, що приводить до виразу

$$\frac{\partial^2\mathbf{J}}{\partial\mathbf{U}^2}(k)(\mathrm{U}(k+1)-\mathrm{U}(k))=-\frac{\partial\mathbf{J}}{\partial\mathbf{U}}(k)\ , \qquad (\Gamma.3)$$

де

$$\frac{\partial^2 \mathbf{J}}{\partial \mathbf{U}^2}(k) = \mathbf{A}\ ; \qquad -\frac{\partial \mathbf{J}}{\partial \mathbf{U}}(k) = \mathbf{b}\ ; \qquad \mathbf{U}(k+1) - \mathbf{U}(k) = \mathbf{x}\ .$$

У даній формі рівняння (Г.3) може бути вирішене за допомогою двох процедур, описаних в [78]: процедури LU декомпозиції, ludcmp і процедури рішення системи лінійних рівнянь lubksb. Після того, як обчислений вектор $\mathbf{x}$. Процедура повторюється до тих пір, поки зміна кожній компоненти $\mathbf{U}(k+1)$ не стає меншим деякого ε. При рішенні системи відносно $\mathbf{x}$ для кожної ітерації алгоритму Ньютона-Рафсона необхідне обчислення кожного елементу якобиана і гессиана. Елемент якобиана з індексом h має наступний вигляд:

$$\frac{\partial J}{\partial u(n+h)} = -2\sum_{j=N_1}^{N_2}[y_m(n+j) - y_n(n+j)]\frac{\partial y_n(n+j)}{\partial u(n+h)} +$$

$$+2\sum_{j=1}^{N_u}\lambda(j)[\Delta u(n+j)]\frac{\partial \Delta u(n+j)}{\partial u(n+h)}\ , \qquad h = 1,\ldots,N_u.$$

Якщо розписати вираз $\frac{\partial \Delta u(n+j)}{\partial u(n+h)}$ і скористатися символом Кронекера дельта функції, яка визначається як

$$\delta(h,j) = \begin{cases} 1 & \text{если} \quad h = j, \\ 0 & \text{если} \quad h \neq j, \end{cases}$$

отримаємо

$$\frac{\partial u(n+j)}{\partial u(n+h)} - \frac{\partial u(n+j-1)}{\partial u(n+h)} = \delta(h,j) - \delta(h,j-1)\ .$$

Елемент з індексами $m,\ h$ гесиана визначається таким чином:

$$\frac{\partial^2 J}{\partial u(n+m)\partial u(n+h)} =$$

$$= 2\sum_{j=N_1}^{N_2}\left\{\frac{\partial y_n(n+j)}{\partial u(n+m)}\frac{\partial y_n(n+j)}{\partial u(n+h)} - \frac{\partial^2 y_n(n+j)}{\partial u(n+m)\partial u(n+h)}[y_m(n+j) - y_n(n+j)]\right\} +$$

$$+2\sum_{j=1}^{N_u}\lambda(j)\left\{\frac{\partial \Delta u(n+j)}{\partial u(n+m)}\frac{\partial \Delta u(n+j)}{\partial u(n+h)} + \Delta u(n+j)\frac{\partial^2 \Delta u(n+j)}{\partial u(n+m)\partial u(n+h)}\right\},$$

$$h = 1,\ldots,N_u, \quad m = 1,\ldots,N_u .$$

Можна скористатися також визначенням дельта-функції для виразу

$$\frac{\partial \Delta u(n+j)}{\partial u(n+m)}\frac{\partial \Delta u(n+j)}{\partial u(n+h)} = \big(\delta(h,j) - \delta(h,j-1)\big)\big(\delta(m,j) - \delta(m,j-1)\big).$$

Вираз $\dfrac{\partial^2 \Delta u(n+j)}{\partial u(n+m)\partial u(n+h)}$ завжди дорівнює нулю.

Остання компоненту, потрібна для отримання $\mathbf{U}(k+1)$, це обчислення виходу об'єкту $y_n(n+j)$ і його похідних. Для цього використовується нейронна мережа.

Г.2 Структура нейронної мережі для моделювання об'єкту

Моделлю об'єкту в алгоритмі NGPC служить нейронна мережа. Початкове тренування нейронної мережі зазвичай виконується в автономному режимі (offline), до застосування управління. Блокова схема для тренування нейронної мережі, що моделює об'єкт, приведена на рис. Г.2.

Нейронна мережа і об'єкт набувають одного і того ж вхідного значення $u(n)$. Нейронна мережа також має додатковий вхід, на який подається або вихідне значенням об'єкту $y(n)$, або нейронній мережі $y_n(n)$. Вибір значення залежить від кроку прогнозу. В процесі тренування мережі її ваги настроюються та-

ким чином, що безліч вхідних значень продукує бажані вихідні значення. Помилка формується між відповідями нейронної мережі $y_n(n)$ і об'єкту $y(n)$. Надалі ця помилка використовується для оновлення вагів за допомогою одного з розглянутих вище методів, наприклад градієнтного спуску [79]. Цей процес повторюється до тих пір, поки помилка не досягне допустимого рівня.

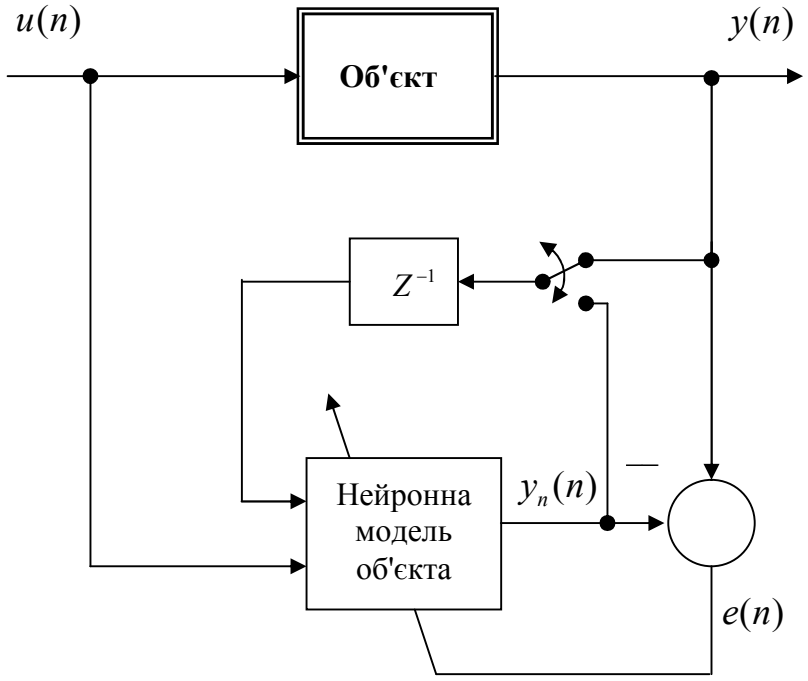


Рисунок Г.2 – Схема тренування мережі в режимі offline

Оскільки для моделювання об'єкту буде використана нейронна мережа, повинна бути розглянута конфігурація архітектури нейронної мережі. Нейронна мережа може бути настроєна або для моделювання входу/виходу, або для моделювання простору станів.

У даній реалізації алгоритму NGPC використовуються моделі, що використовують тільки вхідні/вихідні координати, оскільки моделювання простору станів вимагає вимірювання станів об'єкту, що не завжди можливо. Згідно сказаному вище, мережі МП відрізняються універсальністю при моделюванні різних об'єктів регулювання. У зв'язку з цим подальші дослідження ведуться тільки з цим типом мережі.

МП в даний час – якнайповніше теоретично досліджена мережа. Як указувалося, існуючі теореми стверджують, що МП за умови достатньої кількості нейронів можуть відображати практично будь-які взаємозв'язки. Тому вони часто використовуються для побудови нейромережкких моделей нелінійних об'єктів. Проте перше, з чим стикаються, це питання про те, яку структуру повинна мати мережу. Знаходження оптимальної структури мережі – проблема, яка до цих пір повністю не вирішена. В цьому випадку на допомогу приходить цілеспрямоване моделювання і практичний досвід.

Визначення структури нейронної мережі полягає, по-перше, з визначення входів мережі і, по-друге, з визначення внутрішньої топології мережі (кількість шарів і нейронів). На рис. Г.3 приведена багатошарова нейронна мережа прямого розповсюдження із структурою, що має вузли затримок за часом.

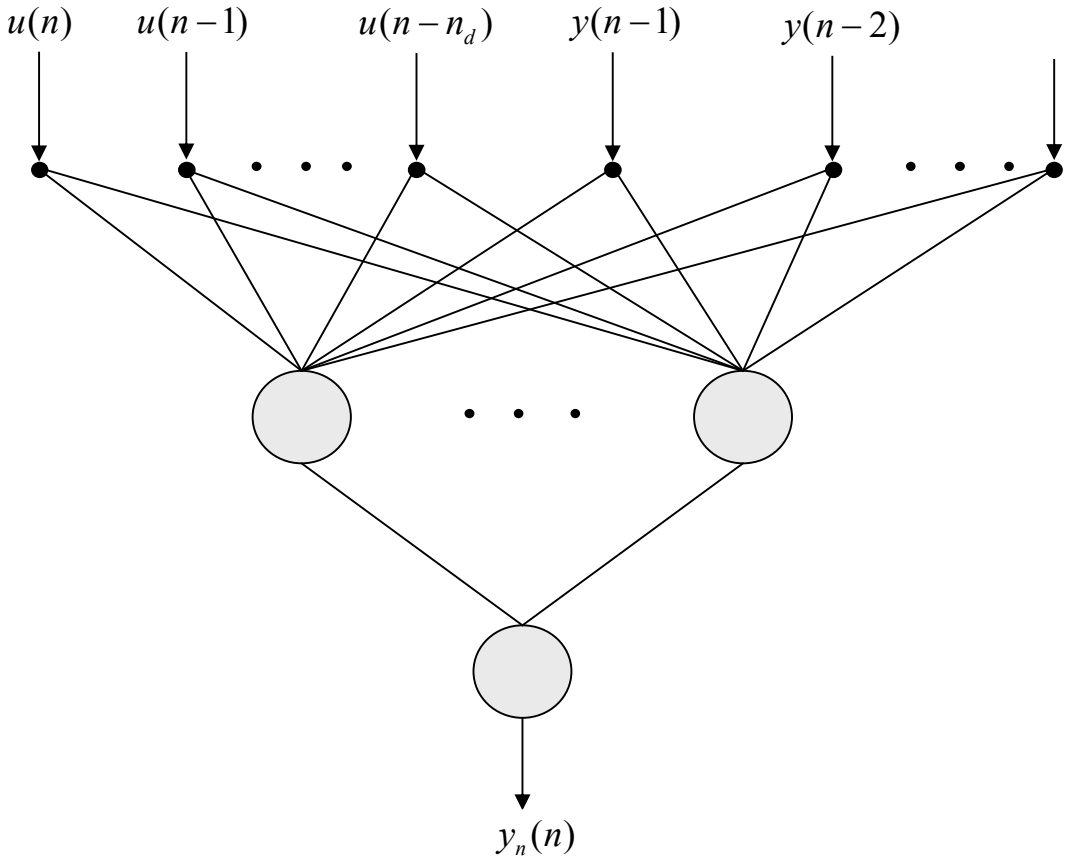


Рисунок Г.3 – Багатошарова нейронна мережа прямого розповсюдження з елементами затримок за часом

На даному прикладі входи мережі складаються із зовнішніх входів $u(n)$ і $y(n-1)$ і відповідних ним вузлів затримок $u(n-1),\ldots, u(n-n_d)$ і $y(n-2),\ldots, y(n-d_d)$. Параметри n_d d_d відображають число вузлів затримки, що відносяться до відповідних вузлів входу. Другий вхід мережі може мати зовнішній вхід $y_n(n)$) і відповідні затримані значення. Мережу має один прихований шар, що складається з декількох прихованих нейронів, які використовують загальну функцію висновку (активації) $f_j(\cdot)$. Вихідний нейрон використовує лінійну функцію висновку з одиничним нахилом для масштабування вихідних значень.

Рівняння мережі з даною архітектурою виглядають таким чином:

$$y_n(n) = \sum_{j=1}^{S^{L-1}} w_j f_j(net_j(n)) + b \qquad (Г.4)$$

і

$$n_j(l) = \sum_{i=0}^{n_d} w_{j,i+1} u(n-i) + \sum_{i=1}^{d_d} w_{j,n_d+i+1} y(l-i) + b_j \ , \qquad (Г.5)$$

де $y_n(n)$ – вихід нейронної мережі;

$f_j(\cdot)$ – функція активації для j-го нейрона прихованого шару;

$net_j(n)$ – аргумент функції активізації j-го нейрона;

S^{L-1} – число нейронів в прихованому шарі;

L – число шарів;

n_d – число вузлів входу, що асоціюються з $u(\cdot)$ без урахування $u(n)$);

d_d – число вузлів входу, що асоціюються з $y(\cdot)$;

w_j – вага, що сполучає j-й прихований нейрон з нейроном виходу;

$w_{j,i}$ – вага, що сполучає i-й вузол входу з j-м прихованим нейроном;

$y(n-i)$ – затриманий вихід об'єкту, використовуваний як вхід для мережі;

$u(n-i)$ – вхід мережі і його затримки;

b_j – зсув j-го прихованого нейрона;

b – зсув нейрона виходу.

Г.3 Прогноз з використанням нейронної мережі

NGPC алгоритм використовує вихід моделі об'єкту для прогнозу динаміки об'єкту для довільного входу від теперішнього моменту часу n до деякого майбутнього моменту $n+k$. Це досягається зрушенням рівнянь (Г.4) і (Г.5) на k маємо

$$y_n(n+k) = \sum_{j=1}^{S^{L-1}} \left\{ w_j f_j(net_j(n+k)) \right\} + b \qquad (Г.6)$$

і

$$n_j(n+k) = \sum_{i=0}^{n_d} w_{j,i+1} \begin{cases} u(n+k-i), & k - N_u < i \\ u(n+N_u), & k - N_u \geq i \end{cases} +$$

$$+ \sum_{i=1}^{\min(k,d_d)} \left(w_{j,n_d+i+1} y_n(n+k-i) \right) + \sum_{i=k+1}^{d_d} w_{j,n_d+i+1} y(n+k-i) + b_j \ . \qquad (Г.7)$$

Особливості рівняння (Г.7) пов'язані з рекурсивним характером прогнозу. Перша сума в (Г.7) визначається відповідно до приведених умов. Умова $k - N_u < i$ управляє попередніми майбутніми значеннями u до $u(n+N_u-1)$. Умова $k - N_u \geq i$ встановлює входи від $u(n+N_u)$ до $u(n+k)$ рівними $u(n+N_u)$. Друга умова виникає тільки, якщо $N_2 > N_u$. Наступне підсумовування в (Г.7) управляє рекурсивною частиною прогнозу. Підсумовування відповідає зворотному зв'язку по виходу нейронної мережі y_n і виконується k або d_d раз залежно від того, яке значення менше. Остання сума в (Г.7) містить попередні значення y. У наступному підрозділі виводяться похідні від рівнянь (Г.6) і (Г.7) для входу $u(n+h)$.

Г.4 Похідні рівнянь нейронної мережі

Для мінімізації функціонала якості в режимі реального часу за допомогою ітераційного рішення, заснованого на градієнті, необхідно використовувати формули для ефективного обчислення градієнта нейронної мережі.

Для обчислення якобіана і гесиана необхідна перша і друга похідні мережі по вхідному вектору. Елементи якобиана виходять диференціюванням $y_n(n+k)$ з рівняння (Г.6) по $u(n+h)$

$$\frac{\partial y_n(n+k)}{\partial u(n+h)} = \sum_{j=1}^{S^{L-1}} w_j \frac{\partial f_j(net_j(n+k))}{\partial u(n+h)}. \qquad (Г.8)$$

Застосовуючи правило диференціювання складної функції для $\frac{\partial f_j(net_j(n+k))}{\partial u(n+h)}$ маємо

$$\frac{\partial f_j(net_j(n+k))}{\partial u(n+h)} = \frac{\partial f_j(net_j(n+k))}{\partial net_j(n+k)} \frac{\partial net_j(n+k)}{\partial u(n+h)} \qquad (Г.9)$$

де $\frac{\partial f_j(net_j(n+k))}{\partial net_j(n+k)}$ – похідна функції виходу і

$$\frac{\partial net_j(n+k)}{\partial u(n+h)} = \sum_{i=0}^{n_d} w_{j,i+1} \begin{cases} \delta(k-i,h), & k-N_u < i \\ \delta(N_u,h), & k-N_u \geq i \end{cases} + \qquad (Г.10)$$

$$+ \sum_{i=1}^{\min(k,d_d)} w_{j,n_d+i+1} \frac{\partial y_n(n+k-i)}{\partial u(n+h)} \delta_1(k-i-1).$$

Відмітимо, що в (Г.10) в останній сумі була введена ступінчаста функція δ_1. Це було зроблено, щоб підкреслити, що сума рівна нулю для $k-i<1$, отже доданки, що задовольняють цій умові, не вимагають обчислень. Щоб отримати елементи гесиана, необхідно продиференціювати рівняння (Г.8), (Г.9) і (Г.10) по $u(n+m)$, в результаті маємо

$$\frac{\partial^2 y_n(n+k)}{\partial u(n+h)\partial u(n+m)} = \sum_{j=1}^{S^{L-1}} w_j \frac{\partial^2 f_j(net_j(n+k))}{\partial u(n+h)\partial u(n+m)} \qquad (Г.11)$$

$$\frac{\partial^2 f_j(net_j(n+k))}{\partial u(n+h)\partial u(n+m)} = \frac{\partial f_j(net_j(n+k))}{\partial net_j(n+k)} \frac{\partial^2 f_j(net_j(n+k))}{\partial u(n+h)\partial u(n+m)} +$$

$$+ \frac{\partial^2 f_j(net_j(n+k))}{\partial net_j(n+k)^2} \frac{\partial net_j(n+k))}{\partial u(n+h)} \frac{\partial net_j(n+k)}{\partial u(n+m)}, \qquad (Г.12)$$

і

$$\frac{\partial^2 net_j(n+k)}{\partial u(n+h)\partial u(n+m)} = \sum_{i=1}^{\min(k,d_d)} w_{j,n_d+i+1} \frac{\partial^2 y_n(n+k-i)}{\partial u(n+h)\partial u(n+m)} \delta_1(k-i-1).$$

Рівняння (Г.12) є результатом застосування правила диференціювання складної функції двічі.

Слід відмітити, що нелінійні оптимізації є процесами, що вимагають великих обчислювальних витрат. Алгоритм Ньютона-Рафсона використовується для створення ефективного обчислювального процесу. При написанні коду, що реалізовує даний алгоритм, всі значення, пораховані в одній процедурі, передаються іншим процедурам для уникнення повторення розрахунку. Найбільш важлива частина коду – обчислення гесиана. Змінні $\frac{\partial net_j(n+k)}{\partial u(n+h)}$ і $\frac{\partial y_n(n+k)}{\partial u(n+h)}$ обчислювані для якобіана, також використовуються для обчислення гесиана, тому вони передаються в процедуру обчислення гесиана. Оскільки гесиан симетричний, тільки верхньотрикутна частина матриці підлягає визначенню. Використання даних двох зауважень дозволяє значно понизити час завантаження процесора.

Таким чином, узагальнене нейроуправління з прогнозом з використання оптимізаційного алгоритму Ньютона-Рафсона для мінімізації функціонала якості дозволяє отримати високу якість управління у разі складних нелінійностей в об'єкті і дає можливість застосування алгоритму в режимі реального часу.

ДОДАТОК Д
ФАЙЛ ВИХІДНИХ ДАНИХ

```
clear;clc;echo on;
% Система управління електроприводом
% механізму підйому мостового крана
% 2025 рік

% Вихідні дані
kd=0.14;
CF=1/kd;
In=92;
Mn=In/kd;
ktp=49.3;
kn=0.05454;
kc=0.104;
Tmc=0.0026;
Ta=0.017;
Te=0.014;
Re=0.1828;
Jd=8.4;
Jm=0.11;
Js=Jd+Jm;
b12=0;
w0=3;
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);

% Матрицы 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;
D=zeros(9,2);
```